

# **TOWARDS BETTER PERCEPTION OF URBAN INFORMATION: A VISUALIZATION PERSPECTIVE.**

by

**QIAOMU SHEN**

A Thesis Proposal Submitted to  
The Hong Kong University of Science and Technology  
in Partial Fulfillment of the Requirements for  
the Degree of Doctor of Philosophy  
in Computer Science and Engineering

October 2019, Hong Kong

# TABLE OF CONTENTS

<b>Title Page</b>	i
<b>Table of Contents</b>	ii
<b>List of Figures</b>	vi
<b>List of Tables</b>	ix
<b>Abstract</b>	x
<b>Chapter 1 Introduction</b>	1
1.1 Motivation	1
1.2 Visualization meets urban information	2
1.2.1 Visualization of human movement (People)	2
1.2.2 Visual assist urban environment exploration (Place)	3
1.2.3 Visual interpretation for techniques (Technology)	3
1.3 Contribution	4
1.4 Proposal organization	5
<b>Chapter 2 Visual Exploration of Human-Scale Urban Forms Based on Street Views</b>	7
2.1 Introduction	7
2.2 Related Work	9
2.2.1 Street View Analysis	9
2.2.2 Urban Data Visualization	10
2.2.3 Multivariate Geographical Data Visualization	10
2.2.4 Comparative Visualization	11
2.3 Background and Analytical Tasks	12
2.3.1 Background	12
2.3.2 Analytical Tasks	13
2.4 System Framework	14
2.5 Data Modeling	16

2.5.1	Data Collection	16
2.5.2	Feature Extraction	17
2.5.3	Data Querying and Filtering	17
2.6	Visualization Design	18
2.6.1	Design Rationales	18
2.6.2	Ranking Explorer	20
2.6.3	AOI Explorer	20
2.6.4	Street Explorer	23
2.6.5	User Interactions	27
2.7	Case Studies	28
2.7.1	City-scale Comparison	28
2.7.2	Region-scale Exploration	29
2.7.3	Street-scale Comparison	31
2.8	Expert Review	32
2.9	Discussion	35
2.10	Conclusion and Future Work	36

### **Chapter 3 Route-Aware Edge Bundling for Visualizing Origin-Destination Trails in Urban Traffic**

3.1	Introduction	37
3.2	Related Work	39
3.3	Problem Statement and System Overview	41
3.3.1	KDEEB Algorithm	41
3.3.2	Problem Identification	43
3.3.3	RAEB Overview	44
3.4	Preprocessing	44
3.4.1	Basic Traffic Concepts	45
3.4.2	Map Matching	45
3.4.3	Hierarchical Route Structure Construction	46
3.4.4	Trail Abstraction	47
3.5	Bundling Method	48
3.5.1	Optimal Kernel Size	48
3.5.2	Density Map Generation	50

3.6 Evaluation	50
3.6.1 Normalized Mutual Information	50
3.6.2 Bundle Deviation	52
3.7 Applications	53
3.7.1 Artificial OD Trails	53
3.7.2 New York Taxi Trips	55
3.7.3 Shenzhen Taxi Trips	56
3.8 Discussion	57
3.8.1 Parameters	57
3.8.2 Performance	59
3.8.3 Applicability and Limitations	60
3.9 Conclusion and future work	61
<b>Chapter 4 Visual Interpretation of Recurrent Neural Network on Multi-dimensional Time-series Forecast</b>	<b>63</b>
4.1 Introduction	63
4.2 Related Work	65
4.2.1 Recurrent Neural Networks	65
4.2.2 Machine Learning Interpretation	66
4.3 Application and Models	67
4.3.1 Application	67
4.3.2 Data Description	68
4.3.3 Models Description	68
4.4 System design	69
4.4.1 Task analysis	69
4.4.2 System Overview	70
4.5 Model Interpretation	71
4.5.1 Relationships between Hidden States and Features	71
4.5.2 Hidden Unit and Feature Clustering	74
4.5.3 Local Feature Importance	76
4.6 Visualization Design	76
4.6.1 Cluster View	77
4.6.2 Feature Importance View	79

4.6.3	Projection View	79
4.6.4	Individual View	80
4.6.5	Interactions and Linkage	82
4.7	Case study	82
4.8	Discussion and Conclusion	83
<b>Chapter 5</b>	<b>Conclusion and Future Work</b>	<b>85</b>
<b>References</b>		<b>87</b>

## LIST OF FIGURES

2.1	Overview of StreetVizor workflow. Our system consists of two phases: data modeling and interactive visual exploration.	15
2.2	Illustration of data preprocessing: sampling locations in New York City are generated from OpenStreetMap (left), a street view image is collected from Google Street View (center), and the image pixels are classified into six features using SegNet (right).	16
2.3	Data model: street views with six-dimensional features of <i>greenery</i> , <i>sky</i> , <i>building</i> , <i>road</i> , <i>vehicle</i> and <i>others</i> , are organized in an octree structure and a street lookup table.	18
2.4	StreetVizor system. (a) Control panel enables multi-scale navigation, ranking exploration, and feature filtering. (b) Side-by-side map views compare the spatial distribution of human-scale urban forms in two areas-of-interest (AOIs). (c) AOI statistic view presents the quantitative measurements, including correlation, histogram, and diversity in the AOIs shown in (b). (d) Street map views present detailed street views along two streets. (e) Street statistic view extends parallel coordinates with street layouts.	19
2.5	The AOI Statistic View combines (a) a scatterplot matrix to show pair-wise correlations between two features, (b) small multiples of histogram bar charts to overview feature distributions, and (c) small multiples of deviation plots to present feature diversity.	22
2.6	Street Map View provides an overview of all street views along a street as colored points, and highlights images on two sides of the street. (a) A tree map showing the feature composition of an image will pop up when the mouse pointer is hovered over the image.	24
2.7	Overview of the construction process for Street Statistic View: (a) Construct minimum bounding box of the street network. (b) Rotate the street network such that its bounding box fits in the rendering space. (c) Plot themeriver style visualization within the rendering space. (d) Enhance parallel coordinates with street layouts on both sides.	25
2.8	AOI Map View compares spatial distributions of human-scale urban forms in Singapore (left) and Greater London (right). Orange points (buildings) are concentrated around the highlighted center area of Greater London, i.e., City of London.	27
2.9	AOI Statistic View in coordination with Fig. 2.8 presents quantitative measurements differences of human-scale urban forms in Singapore (red) and Greater London (blue).	28
2.10	Top three districts with highest <i>building</i> ratios, while bottom three with highest <i>greenery</i> ratios in Hong Kong.	30

2.11	Region-scale comparison of Tanglin in Singapore with Central Park in New York City.	31
2.12	Street Explorer compares the differences in human-scale urban forms of two streets in Brooklyn, New York City (left) and Kowloon, Hong Kong (right). The left street views contain more balanced features, whereas the right street views are dominated by <i>building</i> and <i>road</i> .	33
3.1	Methods for visualizing OD trails in urban traffic: (a) map matching, (b) vector map, (c) & (d) KDEEB bundles rendered in density and direction, respectively.	40
3.2	KDEEB applied to taxi trips in Manhattan with different kernel sizes $p_r$ : (a) 120, (b) 80, (c) 40, and (d) 20. More detailed bundles are generated with smaller $p_r$ .	42
3.3	Overview of RAEB pipeline. Our method mainly consists of three phases: <i>Processing</i> for creating a proper initial layout, <i>Bundling</i> for bundling input OD trails, and <i>Evaluation</i> for generating a stable bundle structure.	43
3.4	Illustration of hierarchical route structure and OD trail abstraction: (a) Three route levels are constructed colored in blue, purple and green, respectively; a raw OD trail is abstracted in accordance to route (b) level-3, (c) level-2, and (d) level-1.	48
3.5	(left) Bundling stability $p_{bs}$ measured at each bundling iteration for different decay ratios $\lambda$ , (right) visually indistinguishable images are generated at iteration 10 and 11 for $\lambda = 0.9$ .	51
3.6	Left: raw 100K artificial OD trails on a hierarchical road network. Right: RAEB bundling result with $p_{ra}$ set to 2.	52
3.7	Leftmost: raw 100K artificial OD trails on a grid road network in there hierarchies. (a) - (d): RAEB bundling results with route awareness parameter $p_{ra}$ set to 0 - 3, respectively.	53
3.8	Density maps of NYC taxi trips: (a) shortest paths mapped onto the road network, (b) KDEEB bundles with kernel size $p_r = 60$ , (c) KDEEB bundles with $p_r = 21$ , and (d) our RAEB bundles with $p_r = 21$ .	53
3.9	Fine scale density maps of NYC taxi trips in Queens zone generated by (a) KDEEB, and (b) RAEB.	54
3.10	Density maps of Shenzhen taxi trips: (a) raw GPS records are mapped onto road network, (b) KDEEB bundles trips on close aterial roads together, while (c) RAEB preserves these roads. All lines are colored according to the OD directions.	55
4.1	RNN Architectures considered in our experiments: A) RNN: the RNN layer is directly connected to the output layer; B) RNN-Dense: adding dense layers between the RNN layer and the output layer.	69
4.2	The system overview. There are three major modules in our system: Preprocessing module, Analysis module and Visualization module.	71
4.3	Compare the response of hidden units(92 and 93) to features ( $PM_{2.5}$ and $SO_2$ ).	73

- 4.4 Cluster score with different cluster number. Left: feature cluster. Right: hidden unit cluster. The horizontal axis represents the cluster number, the vertical axis represents the cluster score. 75
- 4.5 Design of Hidden unit distribution and feature glyph. A) Hidden unit cluster; B) Hidden unit distribution; C) Feature cluster; D) Feature distribution for selected features; E) Feature glyph design; G) Response link MultiRNNExplorer contains multiple coordinated views to support exploring and understanding RNNs' behaviors on multi-dimensional time-series data, especially on hidden unit response and feature importance. The Configuration Panel (B) allows users to select a RNN models and configure parameters. To reveal model mechanism, the Cluster View (A) summarizes the hidden unit clusters' response to feature clusters, and the Feature Importance View (C) summarizes the temporal importance of input features. The Projection View (E) displays a data overview, allowing users to identify and select sequence instances of interest for further analysis. The selected instances will be shown by the Individual View (D). 77
- 4.6 Design of Hidden unit distribution and feature glyph. A) Hidden unit cluster; B) Hidden unit distribution; C) Feature cluster; D) Feature distribution for selected features; E) Feature glyph design; G) Response link 78
- 4.7 Individual design and the alternative designs. A) Individual View. A1) Feature Trend Chart; A2) Cluster Importance Chart; A3) Top Features Chart. B) themeriver as the alternative design of the Cluster Importance Chart; C) and D) node-like sequence and node sequence as the alternative design of Top Features List. 80

## LIST OF TABLES

3.1	Route hierarchy, OSM indicator and hierarchy score	46
3.2	Main parameter adaptions in RAEB, in comparison with these in KDEEB.	57
3.3	Statistics comparison of KDEEB and RAEB for datasets used in the experiments.	59
4.1	There are two types of features taken as input: air pollution and meteorology.	68
4.2	Configuration and performance of RNNs, including vanilla RNN, GRU, LSTM, and the RNNs with dense layer (e.g., RNN-Dense). The performance is evaluated by the mean square error (MSE) of $PM_{2.5}$ ; low MSE represents better performance.	83

# **TOWARDS BETTER PERCEPTION OF URBAN INFORMATION: A VISUALIZATION PERSPECTIVE.**

by

**QIAOMU SHEN**

Department of Computer Science and Engineering

The Hong Kong University of Science and Technology

## **ABSTRACT**

Rapid urbanization has become one of the most important global trends in the last 50 years. Although half of the world's population live in urban areas and contribute to 80 percent of the world's GDP , the ever more crowded urban areas result in a series of problems, such as traffic congestion, pollution, insufficient resources, and unbalanced urban infrastructure. Fortunately, the development of sensing technology has made data collection and processing easier and cheaper, thus providing an opportunity for people to understand the phenomenon or even determine the solutions to address these problems. However, due to the high dimensionality, heterogeneity of the dataset, and the complex analytical tasks, the pure automated techniques are insufficient in the exploration of urban information. On the other hand, the human analysts with sharp perception and domain expertise cannot deal with large volumes of data without powerful tools. Visualization bridges the gap between analysts and automated techniques, and it has been widely applied in the exploration of urban information.

In this proposal, we introduce several novel visual analytics techniques that cover the three domains in urban information exploration: place, people, and technology. In the first work, we

propose StreetVizor, a visual analytics system that helps urban planners to explore fine-scale living environments. The system automatically extracts the features of human-scale urban form from street view images through machine learning techniques. Then, a comprehensive analysis framework and novel visual designs are proposed to support free exploration from multiple levels. In the second work, we target the visualization of massive human movement data. We propose route-aware edge bundling, which visualizes the overview of origin–destination trails. By introducing the additional graph structure as constraints, the trail bundles can follow the traffic network in the city. In the last work, we focus on the model interpretation in the application of air pollutant forecast. We propose MultiRNNExplorer, which visualizes the recurrent neuron network behaviors in multi-dimensional time-series forecast. To validate the effectiveness of the proposed techniques, we conduct several studies based on real-world datasets and domain expert interviews.

# CHAPTER 1

## INTRODUCTION

### 1.1 Motivation

The ongoing urbanization process has been one of the most important trends after World War II . According to the recent urbanization report from the United Nations [17], between 1950 and 2018, the estimated urban population has increased more than fourfold, from 0.8 billion to 4.2 billion. The rapid urbanization has also resulted in a series of problems, such as the traffic congestion and environmental pollution, which attract increased attention from the research field. Traditionally, researchers and urban planners will take tremendous effort to understand and explore large scale urban dynamics due to limited resources and existing cases.

Fortunately, the rapid development of information and communications technology has made urban data collection and analysis cheaper and easier, thus providing opportunities for people to study problems in urban areas and boost the development of relative disciplines, such as urban informatics [9] and urban computing [30]. For example, street view images allow urban planners to conduct urban environment auditing without going to the field. Positional data tracking and acquisition techniques embedded in mobile devices allow researchers to capture the patterns of crowd movement among urban residents. Industrial emissions and meteorological data collected from the monitoring stations help domain experts forecast the air quality accurately and provide control strategies to the government.

Although computers have a great advantage in fast computing and large information storage, fully automated techniques still have limitations in analyzing urban data due to the complexity and variety of analytical tasks in the real world. Final decisions in analyzing data require human intervention, as humans possess acute perception and considerable experience in the domain field. Visualization, which is defined as “the study of transforming data, information, and knowledge into interactive visual representations” [15], bridges the gap between humans and computing techniques.

In this chapter, we first introduce the existing visualization techniques in urban data exploration. Then, we discuss the major contributions of this proposal, followed by the proposal organization.

## 1.2 Visualization meets urban information

The analysis of model urban always refers to the intersection of three domains: people, place and technology [34], which is covered by the proposed work in this proposal.

- **People.** People refers to the residents, citizens, community groups as well as the relative data such as commuting, social network, etc.
- **Place.** Place refers to the physical environment such as the urban sites, locales and habitats which if formed by nature or human activities.
- **Technology.** Technology refers to a variety of techniques span informatics, computing techniques, wearable devices, etc.

Before introducing the contribution of this proposal, we first briefly introduce the existing techniques from these three aspects:

### 1.2.1 Visualization of human movement (People)

As a general topic in visualization discipline, how to effectively visualize the human movement has been long studied in recent years. An overview has been proposed to categorized the existing movement visualization techniques into three classes: **pattern extraction, direct depiction, summarization** [3, 126].

The **pattern extraction** methods leverage effective visualization to discover the mobility pattern from the movement data such as the traffic congestions [123], route assessment [54, 120], commuting patterns [11, 117], co-occurrence [87, 126]. Compare to other two types of techniques, the pattern extract methods aim to solve the specific application problem directly and these techniques are always difficult to extend to other applications.

The **direct depiction** directly plot the movement data according to the data format: such as the line segments for OD trails, and polylines for trajectories [3, 33, 63]. Such techniques preserve all

the information of individual movement. However, when the dataset is large, the visual clutter the render workload will be the major limitation which hinders the analysts' ability to explore the data.

The **summarization** techniques aim to provide an overview of movement data. Most of these summarization techniques conduct statistical calculation and aggregation before rendering. The typical visualization can be heatmap [124], density map [68], flow map [45], matrix [125] and node-link [118]. Due to the ability to present the high-level perception and alleviate the data uncertainty [3], the summarization techniques are widely used in movement visualization.

### **1.2.2 Visual assist urban environment exploration (Place)**

The rapid urbanization cause a series of changes of living environment including the pollutant and urban forms. Visualization techniques has been used in these domain to discover the potential problems and support the decision making. Since the different application have significantly different analysis requirement, most of the existing work target at specific tasks.

By combining the data mining and cluster visualization, HydroQual [1] is proposed to support a visual analysis of river quality. To reason the air pollutant in Hong Kong, Qu et al. [95] propose a visual analytics system consists of several novel design including circular pixel bar chart and a parallel coordinates with S-shape axis. Further more, Deng et al. [25] propose a visual analytics system assisting the air quality analysts in exploring the air pollutant propagation patterns.

Visual analytics also enable the urban planers to design and evaluate the city construction. For example, Ferreira et al. [31] propose Urbane, a 3D multi-resolution framework, which enable the urban planers to conduct the urban development in a data-driven way. Zeng et al. [132] proposal a tool to facilitates the urban vitality. Such tool presents both of the physical entities and urban design metrics and allow the expert to quickly discover the city blocks which could be improved.

### **1.2.3 Visual interpretation for techniques (Technology)**

The computing techniques play an important role in the exploration of urban information. Even though more and more techniques are adapted to urban data exploration, most of them are sill used as a black box. For most of the technique developers and users, the interpretability is an important property of the techniques in a variety of real world applications. The interpretation can help the

technique developers to improve the technique performance and help the technique users improve their confidence [106].

The technique interpretation can be classified into two categories: model reduction and feature contribution. Model reduction methods usually learn a surrogate model to approximate the original complex model. The surrogate model is usually simple and interpretable, such as linear regression [96] and decision trees [23]. Depending on the ways of approximating the original model’s behaviors, there are three main ways to conduct model reduction: decompositional, pedagogical, and eclectic [2].

Feature contribution methods help users understand the relationships between input features and the output prediction. They usually assign each feature an importance score to indicate how it impacts the final prediction. One classical work is Partial Dependence Plot (PDP) [35], which depicts how feature value changes affect predictions. A PDP is usually visualized as a line chart in which the x-axis represents feature values and the y-axis represents prediction possibilities (partial dependence scores). For each feature, its partial dependence scores are usually calculated by iteratively fixing the feature input of all the data points to a certain value and getting the average prediction. One recent work, SHAP [82], also calculates feature attribution, but from a local perspective.

Even though variety of techniques has been proposed to interpret the mechanism of the complex techniques, very few of them target at the application of the urban information exploration which always take large amount, high-dimensioanl and complex data as input.

### 1.3 Contribution

In this proposal, we introduce several visualization techniques in assisting urban data exploration tasks as well as the understanding of urban computing techniques:

- **Visual Exploration of Human-Scale Urban Forms.** We propose StreetVizor, an interactive visual analytics system that helps planners leverage their domain knowledge in exploring human-scale urban forms based on street view images. Our system presents two-stage visual exploration: 1) an AOI Explorer for the visual comparison of spatial distributions and quantitative measurements in two areas-of-interest at city- and region-scales; 2) and a Street

Explorer with a novel parallel coordinate plot for the exploration of the fine-grained details of the urban forms at the street-scale. We integrate visualization techniques with machine learning models to facilitate the detection of street view patterns.

- **An edge bundling technique for visualizing origin-destination trails in urban traffic.** We introduce RAEB(route aware edge bundling), a novel edge bundling technique to visually summary the OD trails in urban traffic data. We identify inconsiderate settings of conventional kernel density estimation edge bundling (KDEEB) when applied to urban traffic data, including non-optimal kernel size and road neglect. The limitations are addressed in RAEB by introduction of a comprehensive pipeline comprising preprocessing, bundling and evaluation processes. A series of new parameters, together with adaptions of existing ones, are employed in the pipeline.

- **Understanding recurrent neural networks on multi-dimensional time-series forecast.**

We propose MultiRNNExplorer, a visual analytics system to interpret RNNs on multi-dimensional time-series forecasts. Specifically, to provide an overview to reveal the model mechanism, we propose a technique to estimate the hidden unit response by measuring how different feature selections affect the hidden unit output distribution. We then cluster the hidden units and features based on the response embedding vectors. Finally, we propose a visual analytics system which allows users to visually explore the model behavior from the global and individual levels.

## 1.4 Proposal organization

The rest of this proposal is organized as follows:

Chapter 2 introduces the design of Streetvizor: specific analysis requirements are described by a collaborating urban planner, and the designs are evaluated and refined against requirements. We illustrate the applicability of our approach with case studies on the real-world datasets of four cities, i.e., Hong Kong, Singapore, Greater London and New York City. Interviews with domain experts demonstrate the effectiveness of our system in facilitating various analytical tasks.

Chapter 3 presents the details about the implementation of RAEB(route aware edge bundling), a new technique to visualize the origin-destination trails in traffic data by edge bundlings techniques. In the evaluation, we conducted experiments with artificial and real-world traffic data to

demonstrate the advancements of RAEB in various applications, including the preservation of road network topology and support of multi-scale exploration.

Chapter 4 proposes a MultiRNNExplorer, a visual analytics system to interpret the recurrent neural networks on multi-dimensional time-series forecast. We introduce the system design and demonstrate the effectiveness of the proposed technique with case studies of air pollutant forecast applications.

Chapter 5 concludes the proposal and discusses possible future work.

## CHAPTER 2

# VISUAL EXPLORATION OF HUMAN-SCALE URBAN FORMS BASED ON STREET VIEWS

Human-scale urban forms indicate urban environments that human can see, smell and touch in their daily lives. The study of urban forms can help planners develop high-quality urban spaces through evidence-based design. Traditionally, the analysis suffers a lot of challenges including the inefficient data collections, and lack of quantitative analysis methods. In this chapter, we present StreetVizor, an interactive visual analytics system that helps planners leverage their domain knowledge in exploring human-scale urban forms. We integrate visualization techniques with machine learning models to facilitate the detection of street view patterns. Our system presents two-stage visual exploration: 1) an AOI Explorer for the visual comparison of spatial distributions and quantitative measurements in two areas-of-interest (AOIs) at city- and region-scales; 2) and a Street Explorer with a novel parallel coordinate plot for the exploration of the fine-grained details of the urban forms at the street-scale. We illustrate the applicability of our approach with case studies on the real-world datasets of four cities, i.e., Hong Kong, Singapore, Greater London and New York City. Interviews with domain experts demonstrate the effectiveness of our system in facilitating various analytical tasks.

### 2.1 Introduction

Human-scale urban form describes fine-scale characteristics of urban environments that can be directly seen, touched, and experienced by a city’s residents in their daily lives [80]. It is typically measured in high-resolution by sight and hearing, i.e., from several to tens of meters. Compared with a city’s scale, which is usually measured in kilometers, this scale is human-oriented. As humans pay more attention to interactive surroundings [38], understanding human-scale urban forms is essential for urban planners in designing high-quality urban spaces. However, traditional urbanism theories, such as small-scale surveys and mapping, are hard to provide in-depth guidance for effective urban planning and design at this fine scale.

Given the advancement of various sensing technologies, e.g., cameras and GPS devices, we can now quantitatively measure human-scale urban forms by analyzing big urban data. In particular, services, such as Google Street View (GSV) [7], provide detailed panoramic views of urban space from different geographic positions. These panoramic views can be utilized to measure various features, including greenery coverage and sky visibility, of human-scale urban forms visible to human eyes. Some pioneering studies have shown that neighborhood environment [98], street-level greenery [73], and even street safety [86] can be precisely assessed from these views.

However, GSV image exploration mainly focuses on either a particular feature (e.g., greenery coverage [73]) or a small area (e.g., neighborhood [98] and street [73, 86]). This deficiency limits its applicability in urban planning, where planners need to 1) quantitatively measure multivariate features of urban forms, including not only greenery coverage, but also sky visibility, and vehicle density [79]; 2) systematically explore urban forms in areas-of-interest (AOIs) at multiple scales, i.e., from small (e.g., streets) to mid (e.g., districts) to large scales (e.g., cities) [78]. In addition, direct means for the comparison of urban forms in two AOIs is desirable to allow planners to utilize information for the quick identification and improvement of factors that affect the quality of urban space.

A visual analytics tool is necessary to fulfill these requirements because it can integrate powerful computing capabilities to quantitatively measure multivariate features, with interactive visual interfaces to systematically explore and compare features in AOIs on demand [107]. Developing such a tool requires considerable effort because of the following reasons: first, as cities comprise vast amounts of street views, an efficient feature extraction algorithm is required to automatically uncover human-scale urban forms. Second, the development of a tool for the visual comparison of multivariate features in two AOIs requires an effective visual design that tackles the challenges of spatial, multivariate, and comparative data visualizations.

In this chapter, we introduce StreetVizor, a visual analytics system for the exploration of human-scale urban forms based on GSV images. We develop the system in an iterative design process: specific analysis requirements are described by a collaborating urban planner, and the designs are evaluated and refined against requirements. To present information in concisely, StreetVizor combines a set of well-established visualization techniques, including coordinated multiple views (CMVs) and scatterplot matrix, with a new design of parallel coordinates that integrate street layout information. Our system utilizes advanced clustering models to enable the efficient exploration of street

view patterns. We apply StreetVizor in real-world datasets containing  $\sim$ 1.7 million of GSV images of four cities: Singapore, Hong Kong, Greater London, and New York City, and demonstrate its effectiveness through interviews with domain experts.

The main contributions of this work include:

- A fully automatic approach measuring human-scale urban forms by applying deep learning techniques on GSV images;
- A visual comparison framework for exploring human-scale urban forms on city-, region-, and street-scales;
- A novel visual design of parallel coordinates that integrate street layout information;
- Interesting insights revealed from case studies and expert interviews, such as the negative correlation between *greenery* and *building* features, and the differences in street views of two cities.

## 2.2 Related Work

This section discusses previous studies closely related to our work.

### 2.2.1 Street View Analysis

GSV system provides high quality and accurate panoramic images of hundreds of cities [7]. In recent years, researchers studying human-scale urban forms have utilized GSV images as a new and convenient data source. For example, researches have shown that the analysis of GSV images can be used to audit neighborhood environments [98], quantify street greenery [73], and predict street safety [86]. Nonetheless, the majority of these studies face scalability issues given their focus on either a particular feature [73] or a small area [73, 86, 98]. These issues can be addressed by incorporating deep learning techniques, which can be used to summarize city landscapes [28] and estimate the demographic makeup of a country [37].

In this work, we collect  $\sim$ 1.7 million GSV images of four representative mega-cities, and apply a deep learning technique [10] to automatically extract desired urban forms from the collected

images. More importantly, we develop an effective visual analytics tool for urban planners to explore human-scale urban forms.

## 2.2.2 Urban Data Visualization

Vast amounts of urban data, including traffic [33, 123], social media [17, 128], environment [32], and simulated urban spaces [115], have been collected in an urban context. Big urban data brings in unprecedented opportunities for evidence-based urban design, and visualization systems can assist domain experts in finding evidence from the data. A systematic overview of visualization systems can be found in [134].

Qu et al. [95] presented a comprehensive visualization system for the analysis of a city’s air pollution that affects the daily lives of residents. Their system integrates parallel coordinates and scatterplots to show relationships between high-dimensional air pollutants. In addition to air pollution, landmark visibility is related to the daily experience of a city’s residents. Ortner et al. [90] visually compared the effects of candidate buildings on landmark visibility from various viewpoints. In this system, users can select a series of ranking schemes, and candidate buildings are then automatically sorted. Similar to our present work, Arietta et al. [8] associated visual elements with city attributes, including violent crime rates and housing prices. They developed various prototype visualizations, such as the visual boundary of urban neighborhoods.

Although different data are explored, these visualizations similarly employ CMVs, because urban data typically exhibits both spatial information and multi-dimensional attributes. Our system also adopts this empirical approach. In addition, to address specific domain problems, we develop effective visualization techniques, including a novel parallel coordinates enhanced with street layouts.

## 2.2.3 Multivariate Geographical Data Visualization

Visualizing multivariate data is a hot topic in the visualization field. Numerous conventional approaches to this topic have been developed, and can be classified into two groups: 1) employing visualization techniques, such as parallel coordinates plot (PCP), scatterplot matrix, and star coordinates; and 2) projecting data points onto a two- or three-dimensional visual space that can be directly plotted on a screen, such as multidimensional scaling and principal component analysis.

All these approaches have pros and cons. For example, although PCP presents all dimensional attributes without information loss, it can easily generate visual clutter with big data and pairwise correlations can only be shown on two nearby coordinates [47]. Many improvements have been developed to address these issues. These improvements include edge bundling to reduce visual clutter [53, 138], and hierarchical data clustering and the navigation of resulting structures [36, 133].

When multivariate data is dependent upon locations, the analytical tasks become more complex because geographical information needs to be revealed. Turkay et al. developed *Attribute Signature* [112], which employs a geographical map and small multiples of multivariate attributes to show geographic variability in attribute statistics. Goodwin et al. [40] further explored multivariate geographical data across scales by adopting new designs to show correlation, scale, and geographical information. The frameworks proposed by both studies can be generalized to explore multivariate geographical data.

In this work, human-scale urban forms to be explored are also multivariate geographical data: the features are in six dimensions and they are dependent on locations. We leverage the advantages of scatterplot matrix and PCP for different analytical tasks. Specifically, we employ scatterplot matrix for exploring features at city- and region scales given that it can effectively reveal correlations between all pair-wise features. We also arrange the views in a way similar to *Attribute Signature* [112], i.e., geographical information is presented on maps and multivariate attributes in small multiples. In addition, we develop a novel PCP enhanced with a themeriver plot, which fits better with the analytical task of showing feature variations along street layout at street-scale.

## 2.2.4 Comparative Visualization

Gleicher et al. [39] classified techniques for visual comparison into three categories: 1) Juxtaposition, i.e., presenting objects next to each other. For example, NodeTrix [129] arranges two human brain networks side-by-side. 2) Superposition, i.e., presenting multiple objects on top of one another. Typical examples are time-series line graphs that plot the changes in several variables over time in the same coordinate system. 3) Explicit encoding, i.e., presenting differences or correlations between objects visually. For instance, the bivariate density map is employed in [131] to show the relationship between departure and arrival movements over space. In practice, these techniques are combined to address complex analytical tasks.

Our work adopts juxtaposition that arranges maps of two AOIs/streets side-by-side (Fig. 2.4(b) & (d)), and superposition to compare multivariate features of two AOIs/streets in the same coordinate system (Fig. 2.4(c) & (e)).

## 2.3 Background and Analytical Tasks

In this section, we introduce our research background and summarize the desired analytical tasks.

### 2.3.1 Background

Although the concept of human-scale urban form has only been recently defined [80], its discussions in the context of urban planning has a long history that can be traced back to the 1960s. A series of pioneering studies [38, 57] claimed the positive effects of understanding human-scale urban forms in designing high-quality urban space. Visible human-scale urban forms are particularly important as human beings tend to pay most attentions to surroundings that can be directly seen [38].

Over the past 10 months, we closely worked with a senior researcher (*SR*) in the field of evidence-based urban design – an emerging research topic in urban planning and design. *SR* pointed out that though urban planners have begun to realize the importance and usefulness of street views in analyzing visible urban forms (e.g., [73, 86, 98]), systematic and efficient methods that can facilitate exploration remain lacking. Hence, *SR* proposed the development of an efficient visual analytics tool for exploring human-scale urban forms based on GSV images.

To better understand the problem domain, we conducted several rounds of structured interviews with *SR*. The main analysis criteria are summarized below:

- **Multivariate Features.** As images contain rich information on the urban environment, the first step is to identify the urban forms for analysis. Here, we identify five key features that can reflect the quality and livability of street spaces [38, 57], i.e., *greenery*, *sky*, *building*, *road*, and *vehicle* features. *Greenery* reflects the pleasing greenery view of a street; *sky* and *building* are correlated with the sense of street closure negatively and positively, respectively; and increments in *road* and *vehicle* ratios decrease the willingness of people to walk and street attractiveness.

- **Street View Crawling.** To reveal the surrounding scenes of a street space, street views have to be crawled appropriately: successive images should reflect the continuous change in surrounding scenes. Hence, the distance between two successive views should not exceed a limit that produces discontinuous scenes; meanwhile, it should not be too small, which will cause computing overload. After experimenting with several options, we find 50 meters is a suitable value for the distance between two successive views.
- **Street View Directions.** Although GSV [41] provides 360-degree panorama imagery, only the front and back images in the directions of street headings at sampling locations are required. Side views are not utilized because of the following considerations: First, side views mainly present building facades and street sides and thus cannot correctly reflect other key features of street space, e.g., *road*. Second, side views are partially contained by the front and back images at nearby sampling locations.

To evaluate the effectiveness of our approach, we first experiment with a few representative cities. *SR* suggested Hong Kong, Singapore, Greater London, and New York City: Hong Kong and Singapore are dense cities with high-rise buildings in Asia. Greater London and New York City are well-planned cities in Europe and the US.

### 2.3.2 Analytical Tasks

After identifying the analysis criteria, *SR* further raised a list of questions for our system to address, including: *How are the identified features distributed in an AOI? What are the feature differences between two AOIs? What are the exact views that people can see on a street? Are there any representative views?*

Based on these questions, we compile a list of analytical tasks:

T.1: **Efficient Multi-scale Exploration:** Human-scale urban forms are associated with street views at different locations that can be organized on city-, region- and street-scales. Planners first need an intuitive overview of the identified feature distributions within a city or a region (T.1.1). Next, planners need to explore the details of the urban forms, such as the exact street views, at street level (T.1.2). Effective interactions are required to assist users in navigating across different scales.

- T.2: **Quantitative Measurements:** SR emphasized the importance of quantitative measurements to evidence-based urban design. Here, given that an AOI/street can contain vast amounts of street views, planners should analyze the statistics of identified features, including correlations between features, distributions, and standard deviations (T.2.1). Filtering street views against the values of a specific feature is also important (T.2.2).
- T.3: **Effective Ranking and Comparison:** To help planners quickly narrow down the exploration scope, features among multiple AOIs/streets should be effectively ranked (T.3.1). Areas/streets with certain features of high values can be easily discovered for further exploration. After planners select two AOIs/streets, they need to compare the differences in spatial distributions (T.3.2) and the quantitative measurements (T.3.3) of the urban forms.

## 2.4 System Framework

StreetVizor is a web-based application comprising two major phases, as illustrated in Fig. 2.1. In the data modeling phase, our system automatically collects hundreds of thousands of GSV images at sampling positions in each city generated from OpenStreetMap (OSM) (Section 2.5.1). Then, we classify the pixels of the collected images into 12 classes using SegNet, and extract the desired feature metric from the classification results (Section 2.5.2). Data collection and preprocessing are conducted offline on a high-performance workstation with 12 core 3.40 GHz Intel Core i7-6800K CPU and a GeForce GTX 1080 graphics card. Though enabled with GPU acceleration, the computation still takes several to 20 hours to preprocess images from each city. Then, we construct data structures, including an octree and a lookup table, to facilitate visual exploration, such as spatial query and filtering (Section 2.5.3). The datasets are stored in a back-end MongoDB server with 2.4 GHz Intel Xeon E5-2620 CPU and 64 GB memory.

The interactive visual exploration phase consists of two stages: 1) Our system provides users with a Ranking Explorer that ranks and compares multiple AOIs/streets based on human-scale urban forms. Users can narrow down exploration by selecting two AOIs/streets for detailed comparison. 2) If two AOIs are selected, the system will present an AOI Explorer that compares the differences in human-scale urban forms in two AOIs at city- and region-scales. The AOI Explorer is composed of CMVs, including two juxtaposition map views for spatial exploration and a superposition statistic view for comparing various quantitative measurements, such as feature correlations

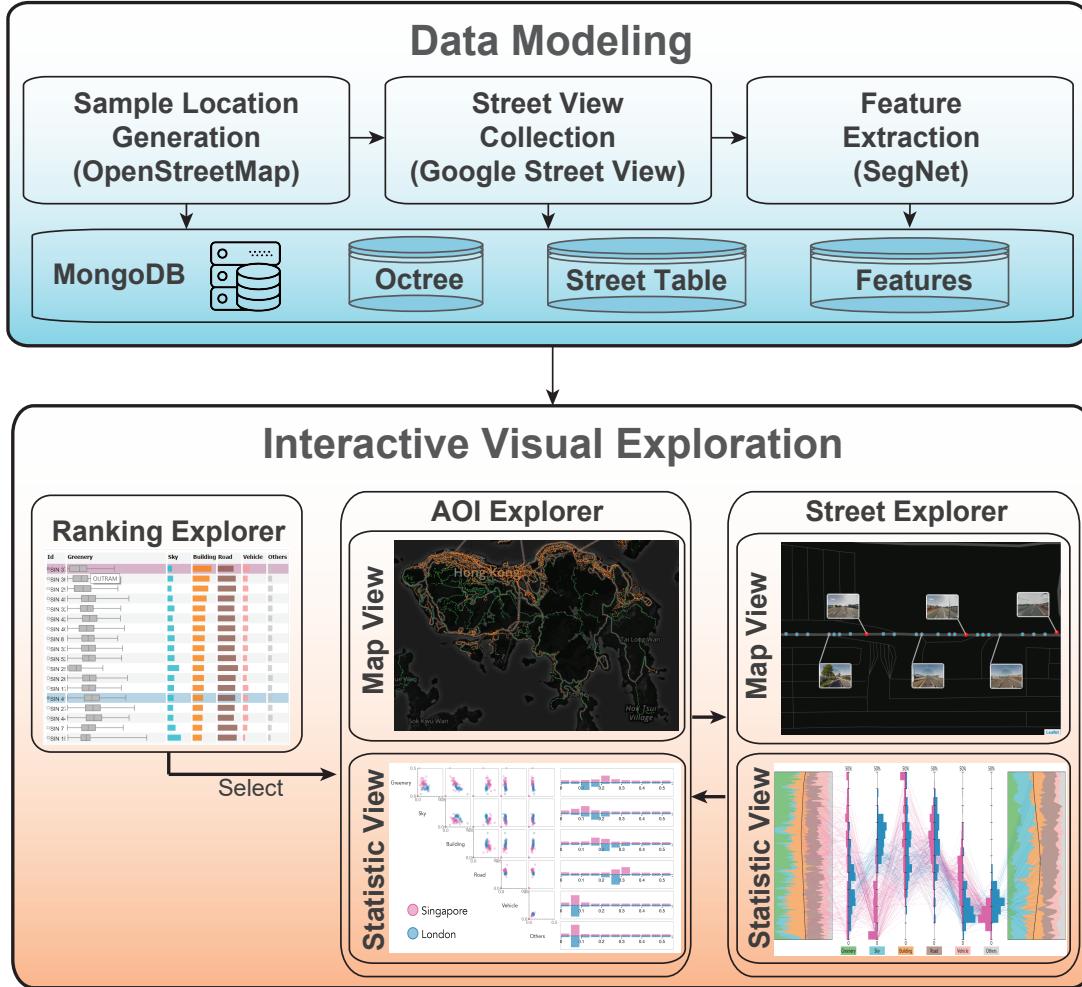


Figure 2.1. Overview of StreetVizor workflow. Our system consists of two phases: data modeling and interactive visual exploration.

and diversities. Users can further navigate down to select two streets, and our system will provide Street Explorer, which presents the fine details of human-scale urban forms at street-level. In Street Explorer, we present map views that show the geographical information and representative images of two streets. We also develop a novel PCP enhanced with themeriver along street layouts, allowing users to compare multivariate features and reveal feature distributions along the two streets. The visualization modules are implemented in D3.js and Three.js for different rendering requirements, and they are integrated using Vue.js.



Figure 2.2. Illustration of data preprocessing: sampling locations in New York City are generated from OpenStreetMap (left), a street view image is collected from Google Street View (center), and the image pixels are classified into six features using SegNet (right).

## 2.5 Data Modeling

In this section, we first describe the collection of GSV images and the extraction of human-scale urban forms from the collected images. Furthermore, we present the methods for data querying and filtering.

### 2.5.1 Data Collection

Based on the configurations defined in Section 2.3.1, we develop an automatic approach to collect GSV images. We first download the area of a city from OSM [88] and extract the road network from the OSM data. Next, we apply a flood-fill algorithm that recursively goes through the entire road network every 50 meters, starting from a randomly selected location. After this operation is completed, a list of sampling locations ( $\{pos\}$ ) with geographic information ( $lat$  &  $long$ ) is generated. We then pass each ( $lat$  &  $long$ ) into GSV API, and extract the corresponding street information ( $SI$ ), including street name ( $s\_name$ ) and heading ( $h$ ). Finally, we download front and back images at each sampling position by passing  $lat$ ,  $long$  &  $h$  with the default field of view and pitch values into GSV API. Together with the downloaded image ( $Img$ ), we model urban forms at human-scale ( $UF_{hs}$ ) at each sampling location as:

$$UF_{hs} := \langle pos, SI, Img \rangle \quad (2.1)$$

We have collected  $\sim 147$  k,  $\sim 183$  k,  $\sim 685$  k and  $\sim 637$  k images of Hong Kong, Singapore, Greater London and New York City, respectively. Fig. 2.2 (left) presents all sampling locations (colored dots) in New York City generated from OSM, and (center) shows a sample street view image downloaded from GSV.

## 2.5.2 Feature Extraction

After collecting street views, we first classify the image pixels into 12 classes (e.g., sky and building) using SegNet [10], which is a robust pixel-wise semantic labeling tool with a global accuracy of 82.8%. Among the 12 classes, we count the number of pixels for the identified five features, i.e., *greenery*, *sky*, *building*, *road*, and *vehicle*, and summarize the remaining pixels as *others*. We then normalize the feature data because pixel counts ( $PC$ ) as raw output values are not intuitive. The normalization is straightforward for each feature value ( $FV$ ):  $FV_i = PC_i / PC_{Img}$ , where  $i \in \{g, s, b, r, v, o\}$  represent the five features and *others*. Hence, all feature values are in the range of [0, 1]. Then, we can replace the image ( $Img$ ) with a feature metric ( $FM$ ) as:

$$Img \rightarrow FM := < FV_g, FV_s, FV_b, FV_r, FV_v, FV_o > \quad (2.2)$$

Fig. 2.2 (right) shows the classification result of the street view image in the center produced by SegNet.

## 2.5.3 Data Querying and Filtering

Based on Equations 2.1 and 2.2, we model human-scale urban forms ( $UF_{hs}$ ) with the following attributes: position ( $pos$ ), street information ( $SI$ ), and feature metric ( $FM$ ). By nature, the data exhibits the following properties: 1) *spatial*, i.e., positions; 2) *multi-scale*, because the positions can be hierarchically grouped in accordance with city and regional units, or street information; and 3) *multivariate*, i.e., the feature metric is in six-dimensional data space.

These complex data natures bring in challenges for our analytical tasks. To address these challenges, we further identify the following querying and filtering models that our system should support:

- **Spatial Query:** To overview the feature distributions within an AOI (T.1.1), our system should first support an efficient query of a list of  $\{UF_{hs}\}$  with their  $pos$  laying in a given AOI. The AOI can be either an administrative zone (e.g., a city or a district) or a user-specified region defined using a lasso tool. We achieve the efficient spatial query operation by organizing all  $\{UF_{hs}\}$  in a city in a four-level octree structure, in which the topmost level is the boundary of each city.

- **Street Query:** To support the exploration of human-scale urban forms at street-scale (T.1.2), our system should allow users to interactively query a street by its name. Here, we create a lookup table with street names as keys, and store corresponding  $UF_{hs}$  ids in each street slot.
- **Feature Filtering:** To accelerate filtering against a particular feature (T.2.2), our system first sorts all  $\{UF_{hs}\}$  to be explored in increasing order for every feature. Then, we adopt a binary search approach in run time.

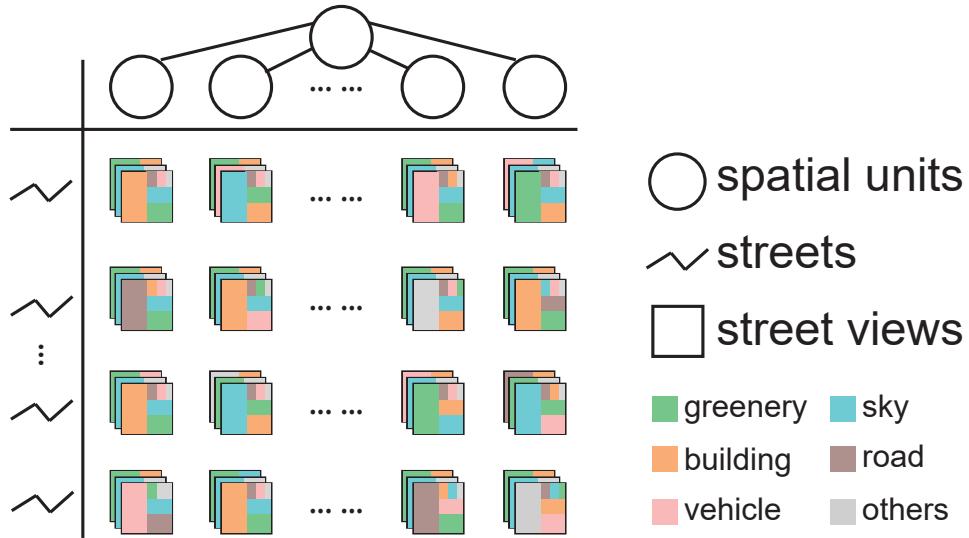


Figure 2.3. Data model: street views with six-dimensional features of *greener*, *sky*, *building*, *road*, *vehicle* and *others*, are organized in an octree structure and a street lookup table.

Fig. 2.3 illustrates the data model that organizes the street views in an octree structure and a street lookup table. Each street view contains the six-dimensional features of *greener*, *sky*, *building*, *road*, *vehicle*, and *others*. We store these querying and filtering models are stored in a back-end MongoDB database, as shown in Fig. 2.1.

## 2.6 Visualization Design

In this section, we first discuss the rationales behind our visualization design. Then, we provide a detailed description of the visualization techniques implemented in our system.

### 2.6.1 Design Rationales

To address the complex analytical tasks (Section 2.3.2), a proper visual design should follow the design rationales below:

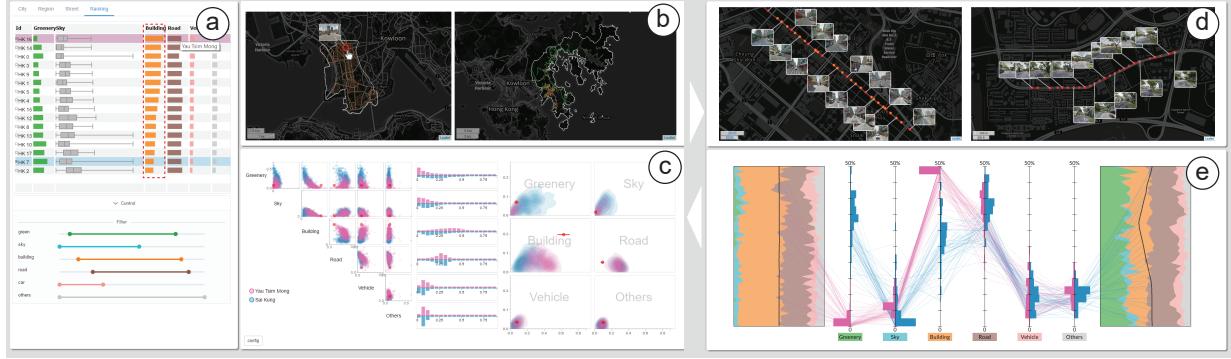


Figure 2.4. StreetVizor system. (a) Control panel enables multi-scale navigation, ranking exploration, and feature filtering. (b) Side-by-side map views compare the spatial distribution of human-scale urban forms in two areas-of-interest (AOIs). (c) AOI statistic view presents the quantitative measurements, including correlation, histogram, and diversity in the AOIs shown in (b). (d) Street map views present detailed street views along two streets. (e) Street statistic view extends parallel coordinates with street layouts.

- R.1: **Overview+Details:** To facilitate multi-scale exploration, our system should follow the information-seeking mantra: “Overview first, zoom and filter, then details on demand” [104]. First, the system should provide an overview of human-scale urban forms at city- and region-scales, and then allow users to explore more details at street-scale. Efficient query and filtering should be provided to enable smooth transitions between these scales.
- R.2: **Coordinated Multiple Views:** Our system should effectively reveal multiple perspectives information of human-scale urban forms, including geographical locations and multivariate features. CMVs that present linked information and allow users to explore data from multiple joint-perspectives fulfill this requirement.
- R.3: **Effective Comparison:** To enable effective data comparison, different comparative visualization techniques should be employed for multiple-perspective information. Specifically, given that spatial information for two AOIs/streets is unsuitable for direct overlay, we adopt side-by-side map views. On the other hand, the feature metric can be mapped on the same scope. Therefor, we select a superposition visualization to reveal the differences.
- R.4: **Visual Consistency:** Since multi-scale and multiple-perspective visualizations are to be designed, the system should maintain visual consistency across different visualization modules. We realize visual consistency by 1) applying the same layout, i.e., presenting spatial information on the top and quantitative measurements on the bottom, in AOI Explorer and Street Explorer; 2) employing consistent color mappings. Specifically, we set green, light blue, or

ange, brown, light pink, and gray to represent the features of *greenery*, *sky*, *building*, *road*, *vehicle*, and *others*, respectively. AOIs/streets on the left and right side are colored as red and blue, respectively.

## 2.6.2 Ranking Explorer

Ranking Explorer is developed to overview feature attributes across multiple AOI/street candidates to help users quickly identify AOIs/streets for comparison (T.3.1). The explorer presents each candidate as a row and arranges its multivariate information in seven columns. The first column provides general information, such as city and region/street id. The remaining columns present the six features' mean values as bars, where mean values are normalized and encoded by bar lengths. Clicking the body of a feature column of interest will expand the column as a boxplot to show statistical distributions. The explorer also allows users to sort candidates against a particular feature and the ranking will update correspondingly. Such designs have been well established and evaluated in a previous work [76].

Fig. 2.4(a) ranks all districts in Hong Kong in accordance with *building* feature (red dashed box). As an example, we observe the detailed statistics of the *sky* feature. We select the column for this feature and expand it to boxplot. By comparing the orderings of *greenery*, *sky*, and *building* features, we find *greenery* and *sky* features are correlated, while they are negatively correlated with the *building* feature. This information helps users narrow down the comparison choices to 1) district HK 16 (Yau Tsim Mong) with the highest *building* ratio and 2) district HK 7 (Sai Kung) with low *building* and high *greenery* ratios, as shown in Figs. 2.4(b) & (c).

## 2.6.3 AOI Explorer

AOI Explorer is developed to provide efficient comparison of human-scale urban forms at city- and region-scales (T.1.1). The explorer integrates coordinated multiple views, including:

### AOI Map View

We develop side-by-side map views (Figure. 2.4(b)) to compare the spatial distributions of human-scale urban forms in two AOIs (T.3.2). Each map view consists of: 1) a background map layer

implemented with Leaflet.js to allow users to change map style (e.g. satellite, street, and sport) for different purposes; and 2) a point density map overlaid on top of the background map layer, with points representing street views. The density points are evenly sampled on each street with an upper limit of 10,000 points. Point color corresponds to the maximum feature value in the street view image. A corresponding street view image will pop up when the mouse pointer hovers over a point. Users can select two cities or regions from the navigation panel, or directly manipulate AOIs on the map views with a lasso tool.

Heat map is an alternative design for the point density map. However, in this work, we focus on the simultaneous analysis of multiple features of human-scale urban forms. Compositing these features into one heat map [21] will require redundant user interactions. In addition, sampling positions are generated along the street network. Thus, no street views is collected from many places across a city. In this case, the heat map will generate ambiguity between the two scenarios of 1) no record or 2) low feature values in a region.

## AOI Statistic View

Besides the map view, which enables the comparison of spatial distributions of human-scale urban forms, we develop AOI Statistic View to allow users to compare various quantitative measurements (T.3.3). Nonetheless, each AOI may contain too many street views (up to several hundred thousand) that will overload the rendering process. Moreover, the street views will occlude each other if we simply plot all of them. To address this problem, we cluster street views into groups based on either their administrative units (districts, divisions, and streets) or a mean-shift clustering algorithm [?] that works as follows.

**Mean Shift Clustering.** We cluster urban forms based on their feature metric and geographical positions. Our algorithm works in the following way: given an input list of  $N$  human-scale urban forms  $\{UF_{hs}\}$ , we first normalize the *lat* & *long* attributes of each  $UF_{hs}$  against the boundary of all  $\{UF_{hs}\}$ . Then, we combine the normalized  $lat_{nor}$  &  $long_{nor}$  with the feature metric to construct a new dataset  $X := \{x_1, x_2, \dots, x_N\}$ , where each data point  $x_i$  is in an eight-dimensional space,  $R^8 := <lat_{nor}, long_{nor}, FV_g, FV_s, FV_b, FV_r, FV_v, FV_o>$ . The distance between two data points is measured as their Euclidean distance. We then estimate a bandwidth  $h$  from  $X$ , and apply mean-shift clustering algorithm on  $X$  using a flat kernel. Here, the bandwidth estimation and mean shift

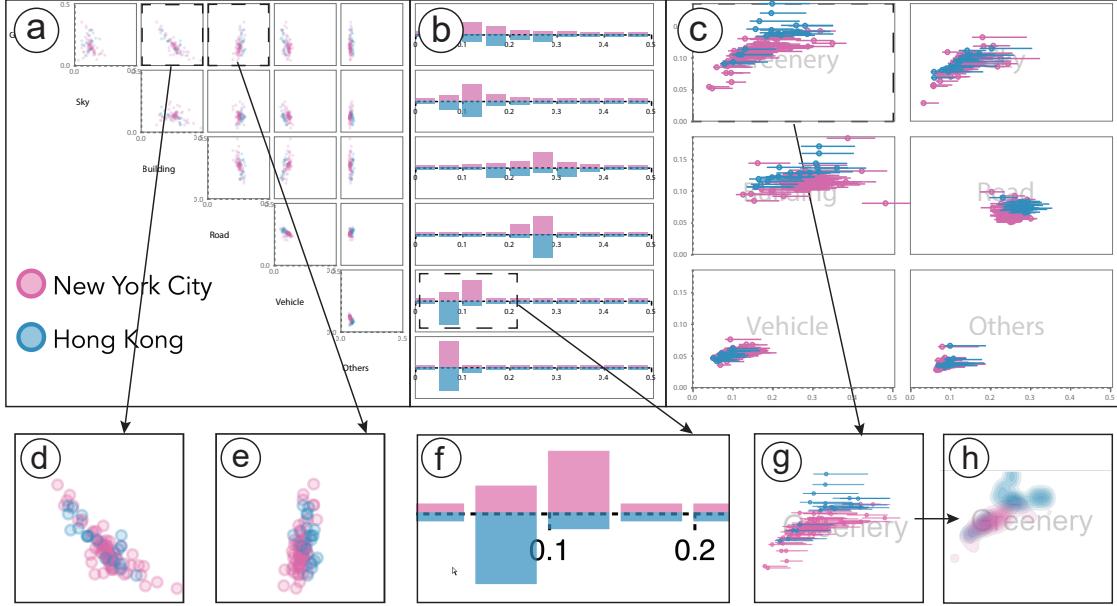


Figure 2.5. The AOI Statistic View combines (a) a scatterplot matrix to show pair-wise correlations between two features, (b) small multiples of histogram bar charts to overview feature distributions, and (c) small multiples of deviation plots to present feature diversity.

clustering are performed with a machine learning library *scikit-learn* [91]. Finally, we generate a list of  $m$  urban form clusters  $C := \{c_1, c_2, \dots, c_m\}$ , where each cluster  $c_i$  contains  $n$  data point  $c_i := \{x_1, x_2, \dots, x_n\}$ .

The clustering process groups geographically close street views with similar feature attributes together. Given that locations are integrated as two-dimensional spaces, the algorithm forms more local clusters than an algorithm without considering spatial information, which usually generates several big clusters with too many street views. The algorithm may be further improved by adopting a network-based distance measurement approach other than Euclidean distance. We expect to form more representative clusters of street views with street network information. Nonetheless, the current approach fulfills our requirement of reducing visual clutter.

After generating the clusters  $C$ , we further compute and visualize the following quantitative measurements:

- **Feature Correlation:** We first compute the mean values of the identified features, i.e., *greenery*, *sky*, *building*, *road*, *vehicle*, and *others* in each cluster  $c_i$ . We then plot the pair-wise correlations using a scatterplot matrix (Fig. 2.5(a)). Notice that the clusters in the left and right AOIs are colored red and blue, respectively.

- **Feature Histogram:** Though feature values fall in the range of [0% - 100%], we seldom find feature values that exceed 50%. Thus, we only consider the range [0% - 50%]. For each feature, we divide the range into 10 even parts, i.e., [0% - 5%), [5% - 10%)..., and aggregate the corresponding value in each street view to each part. Then, histograms for each feature are plotted as bar charts in an up and down manner for AOIs on the left and right, respectively. The six histogram bar charts are arranged in small multiples next to the scatterplot matrix (Fig. 2.5(b)).
- **Feature Diversity:** We use standard deviation to indicate the measured feature diversity measured for each cluster, and design diversity views that are arranged as six side-by-side small multiples for each feature as shown in Figure 2.5(c). In every feature diversity view, each cluster is represented as a dot with its x-value indicating the averaged feature value and y-value indicating standard deviation. In addition, we use line segments to indicate the largest and smallest feature values in each cluster. In case an AOI may have hundreds of clusters that lead to serious visual clutter problem, we implement a contour map view [16] (Fig. 2.5(h)) to show overall distribution patterns. Users can interactively choose one of the views in accordance with different requirements.

The AOI Statistic View comparison of New York City (red) and Hong Kong (blue) is shown in Fig. 2.5. From the view, we can easily identify some obvious patterns. For example, in both cities, *greenery* and *building* features are negatively correlated (Fig. 2.5(d)). The correlations between other pairwise features, e.g., *greenery* and *road* (Fig. 2.5(e)), are not obvious. From Fig. 2.5(f), we find that the vehicle ratio is higher in New York City than in Hong Kong. In addition, we find that although the *greenery* values are similar for both cities, diversity is higher in Hong Kong (Fig. 2.5(g)). This results reflects that greenery is better integrated into street space in New York City than in Hong Kong.

## 2.6.4 Street Explorer

Street Explorer is developed to enable the efficient comparison of human-scale urban forms at street-scale (T.1.2). The explorer adapts the same layout as that in the AOI Explorer, i.e., juxtaposition map views are placed on the top of the explorer, whereas detailed statistics view are on the bottom.



Figure 2.6. Street Map View provides an overview of all street views along a street as colored points, and highlights images on two sides of the street. (a) A tree map showing the feature composition of an image will pop up when the mouse pointer is hovered over the image.

## Street Map View

Similar to AOI Map View, Street Map View is also developed on top of a background map layer that is implemented using Leaflet.js. Street views on the street are over-viewed as points with colors that correspond to primary features, i.e., green for *greenery*, light blue for *sky* and orange for *building*. In contrast to AOI Map View, Street Map View presents more details of human-scale urban forms by displaying the corresponding street view images along the two sides of a street. The images are evenly selected in the direction of the street heading. In particular, the feature compositions of an image are displayed as a tree map [103] when users hover their mouse pointer over the image. An example is provided in Fig. 2.6(a).

## Street Statistic View

The design goal for this view is to allow users to quantitatively compare human-scale urban forms from two streets. Although we can utilize the same views shown in the AOI Statistic View, our collaborating domain expert *SR* is not satisfied. *SR* strongly recommended encoding street layout information in the view, so that users can better leverage their knowledge about streets to perform in-depth analysis. *SR* felt the spatial information was not well integrated with the Street Map View.

Based on this goal, we experimented with a few alternative designs and informally evaluated design prototypes with *SR*. For our first prototype, we designed some glyphs, such as a radial chart

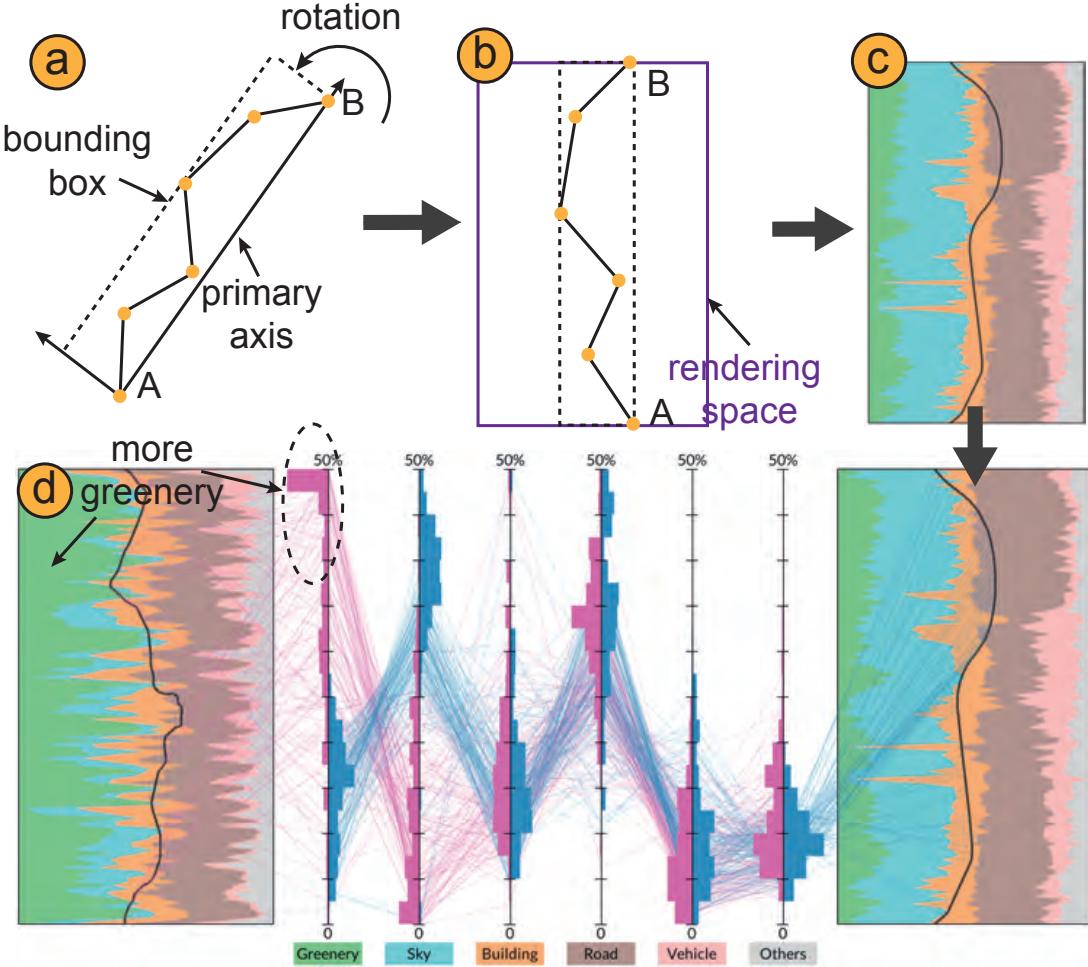


Figure 2.7. Overview of the construction process for Street Statistic View: (a) Construct minimum bounding box of the street network. (b) Rotate the street network such that its bounding box fits in the rendering space. (c) Plot themeriver style visualization within the rendering space. (d) Enhance parallel coordinates with street layouts on both sides.

or tree map, and directly overlaid the glyphs on the map view. The design was aborted because isolating statistics on the two maps weakened the effects of comparison. In addition, such a design easily generated visual clutter, which requires redundant user interactions or clustering to address. For the second prototype, we encoded street layout information in a similar design as the AOI Statistic View. However, this method was extremely difficult because the rendering space was fully utilized.

**PCP with Street Layout.** Inspired by [95], we ultimately develop a design of a parallel coordinates enhanced with street layouts (Fig. 2.7(d)). We elaborate the construction of this design as follows:

*Bounding Box Construction* (Fig. 2.7(a)). The first step is to construct a minimum bounding box (*MBB*) of the street layout. We find the starting (*A*) and ending (*B*) points, and generate a primary

axis pointing from  $A$  to  $B$ .  $MBB$  is generated as the minimum rectangle that contains all nodes of the street network parallel with the primary axis.

*Street Layout Rotation* (Fig. 2.7(b)). Next, we rotate the street layout such that  $MBB$  fits in the rendering space, i.e.,  $A$  and  $B$  lay on the top and bottom (or vice versa) sides of the rendering space, and the rotated  $MBB$  lays in the center. In the case that  $MBB$  is wider than the rendering space, the street layout outside the rendering space is clipped. The rotation direction (clockwise or anticlockwise) is determined based on the direction that produces the minimum rotation angle.

*Themeriver Plotting* (Fig. 2.7(c)). To fully utilize rendering space, we plot a themeriver-style [46] visualization to show changes in feature values along a street layout. Here, the themeriver is plotted in the vertical instead of the traditionally horizontal direction because the street layout is aligned along the y-axis after rotation. Next, given that street views are sampled evenly along the street layout, we map the street views equally onto the y-axis, i.e., y-values of the themeriver plot reflects the relative positions of street views along the street layout.

We start plotting the themeriver visualization using the left side of the rendering space as the baseline and calculate the upper bound x-values for the *greenery* feature. The upper bound x-values of the *greenery* layer are then used as baseline values for the next feature, i.e., *sky*. This process is repeated until all features are plotted. To support better feature comparison, our system allows users to reorder feature sequence by clicking on the feature layer of interest and shifting it to the left side [14].

*PCP Integration* (Fig. 2.7(d)). The themeriver plot can be used as a coordinate for the PCP. To enable comparison between two streets, we integrate their themeriver plots into a PCP on left and right sides, and the identified features are used as other coordinates. Each street view is conveniently presented as a polygonal line, which is colored as red and blue for left and right street views, respectively. In addition, we aggregate the feature values via binning techniques on left and right sides of each feature coordinate to facilitate the visual comparison of feature distributions.

Fig. 2.7(d) presents an example Street Statistic View. A first glimpse at the two themeriver plots, we can find that the left street contains more *greenery*, while on the right street people can see more *sky*. This can be found in the distribution comparison on the *greenery* & *sky* coordinates in the middle. Besides, we can also find that feature values vary dramatically on the left themeriver plot, showing street views are very different along the left street.

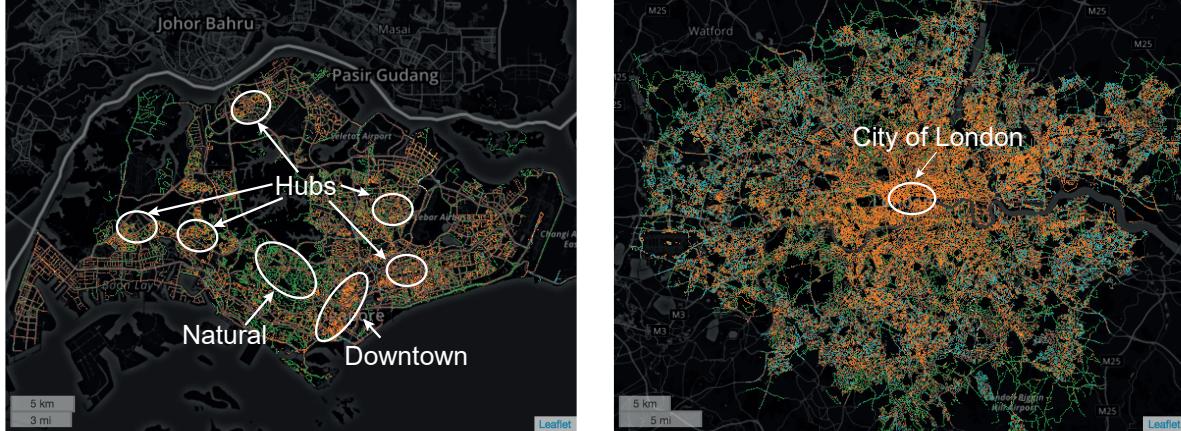


Figure 2.8. AOI Map View compares spatial distributions of human-scale urban forms in Singapore (left) and Greater London (right). Orange points (buildings) are concentrated around the highlighted center area of Greater London, i.e., City of London.

## 2.6.5 User Interactions

In addition to basic interactions in each view, StreetVizor also provides a control panel (Fig. 2.4(a)) that enables:

- *Multi-scale Navigation.* To help users navigate effectively across different scales, we develop city-, region- and street-panels. The city-panel lists all four cities, i.e., Hong Kong, Singapore, Greater London, and New York City. Users start navigation by selecting a city. The region-panel will then list all administrative regions under the city. For instance, City of London and Kingston will be listed if Greater London is selected. Similarly, the street-panel lists all the streets inside a selected city or region.
- *Feature Filtering.* Our system also supports filtering human-scale urban forms against a specific feature by specifying the value range with feature sliders. By default, each feature slide within [0 - 1], and users can change the minimum and maximum values by dragging the slider buttons.

In addition, our system also supports the following user interactions to facilitate visual exploration.

- *Details on Demand.* To enable overview+details (R.1), we develop a set of interactions that allow users to explore the details of human-scale urban forms on demand. For example, if

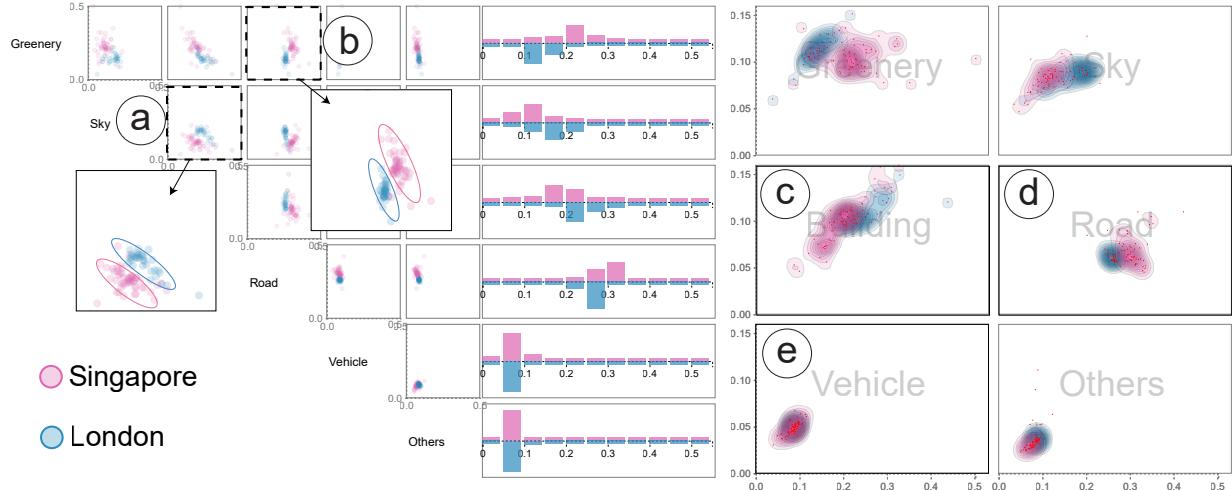


Figure 2.9. AOI Statistic View in coordination with Fig. 2.8 presents quantitative measurements differences of human-scale urban forms in Singapore (red) and Greater London (blue).

a street view point is selected in the AOI Map View, the corresponding image will show up. Thus, users can leverage their domain knowledge by visually examining street views.

- *Linking.* Our system supports automatic linking among the visualization modules in both AOI Explorer and Street Explorer for coordination across multiple views (R.2). For example, if a specific street view on the AOI Map View is selected, the cluster that contains the street view in the scatterplot matrix and the diversity views will be highlighted accordingly.

## 2.7 Case Studies

This section presents three case studies on the application of StreetVizor in assisting urban planners to explore and compare human-scale urban forms on the city-, region-, and street-scales.

### 2.7.1 City-scale Comparison

*Comparing Spatial Distribution.* With the AOI Map View, our system allows users to compare spatial distributions of human-scale urban forms in two AOIs (T.1.1). Fig. 2.8 presents a comparison between Singapore (left) and Greater London (right). As shown in the figure, more orange points surround the highlighted white circle, whereas more green points can be found at marginal areas in Greater London. This indicates that more buildings are constructed around the City of London,

reflecting that this city is more urbanized compared with other areas. By contrast, Singapore’s urban forms are evenly distributed. More orange points are found in the highlighted downtown and region hubs, and more green points can be found in natural areas. Our collaborator *SR* explained the reason for the different spatial distributions of urban forms in Greater London and Singapore: Greater London likely expanded its urbanization from the City of London to surrounding areas, whereas Singapore, as an island city, cannot expand.

*Comparing Quantitative Measurements.* We can further explore the differences in quantitative measurements with AOI Statistic View (T.2.1 & T.3.3). Fig. 2.9 shows the AOI Statistic View in coordination with Fig. 2.8. From the scatterplot matrix, we find that *greenery* and *building* features are negatively correlated. This result is the same as that shown in Fig. 2.5(d). Given that *buildings* are artifacts and *greenery* is natural, the negative correlation between *building* and *greenery* features in all four cities indicates that artifacts increase and natural environments decrease with increasing urbanization. To improve livability, many urban planners have proposed integrating more natural spaces in street space. In addition, Fig. 2.9(a) shows that *sky* and *building* ratios are higher in London than in Singapore. By contrast, *greenery* and *road* ratios are higher in Singapore than in London (Fig. 2.9(b)). These findings can also be found from the middle histogram bar charts. In addition, the diversity views further show some interesting patterns. For example, *building* and *road* diversities are more concentrated in London than those in Singapore, as shown in Fig. 2.9(c) & (d), and likely resulted from the different standards for building and road construction in the two cities. *Vehicle* usage rates are low in both cities, as shown in Fig. 2.9(e).

## 2.7.2 Region-scale Exploration

Given that human-scale urban forms are highly associated with the daily lives of residents, we posit that urban forms could reflect the functionalities of a region. Study 1 already reveals the negative correlations between *greenery* and *building* features, and we hypothesize that these two urban forms can reflect urbanization levels. To evaluate the hypothesis, we further explore these two features at region-scale (T.1.1). Using ranking functions, we sort administrative districts in Hong Kong based on these two features. Fig. 2.10 presents an overview of three districts with highest values for each feature. The top three districts are Yau Tsim Mong, Wan Chai, and Kowloon City. They are all business centers in Hong Kong, and we find these districts are mostly covered by orange points (*building*). By contrast, the bottom three districts contain highest values for *greenery*. In

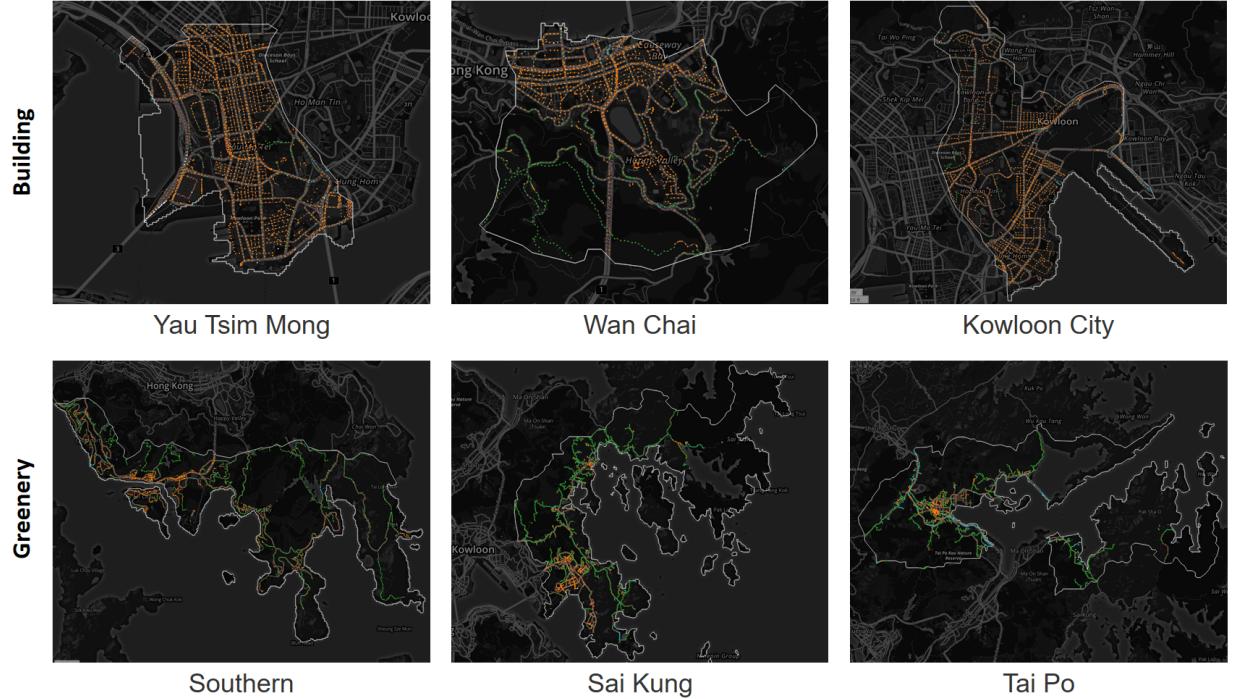


Figure 2.10. Top three districts with highest *building* ratios, while bottom three with highest *greenery* ratios in Hong Kong.

particular, the greenest district is Southern Island, which is reserved as country parks. The other two districts are also well-known natural areas in Hong Kong.

In addition to exploring regions with different functionalities, urban planners are also interested in comparing regions with similar functionalities. Here, we leverage our knowledge of famous regions in two different cities, i.e., Tanglin in Singapore (see the region highlighted as natural in Fig. 2.8) and Central Park in New York City. Both of these regions are well-known parks. After selecting these regions, we find that nearly all streets in the two regions are dominated by green points, indicating that both regions contain considerable *greenery* as intended. However, we find some differences between these two regions by looking deeper into the Street Statistics View (Fig. 2.11). First, through the feature histogram bar charts, we find that Central Park has slightly higher *greenery* ratios (A1), whereas Tanglin has slightly higher *building* ratios (A2). Though these differences are marginal, they reflect that more buildings are present in Tanglin, partially because Singapore tries to maximize land usage for building construction. In addition, by grouping street views based on street units, we glean more information from the diversity plots. Here, we first obtain the overview that *greenery* feature has relatively higher mean values and deviations than the other five features. We also find some anomalies. For example, some streets have relatively low mean values but high deviations for *greenery* in Central Park (B1), whereas certain streets have

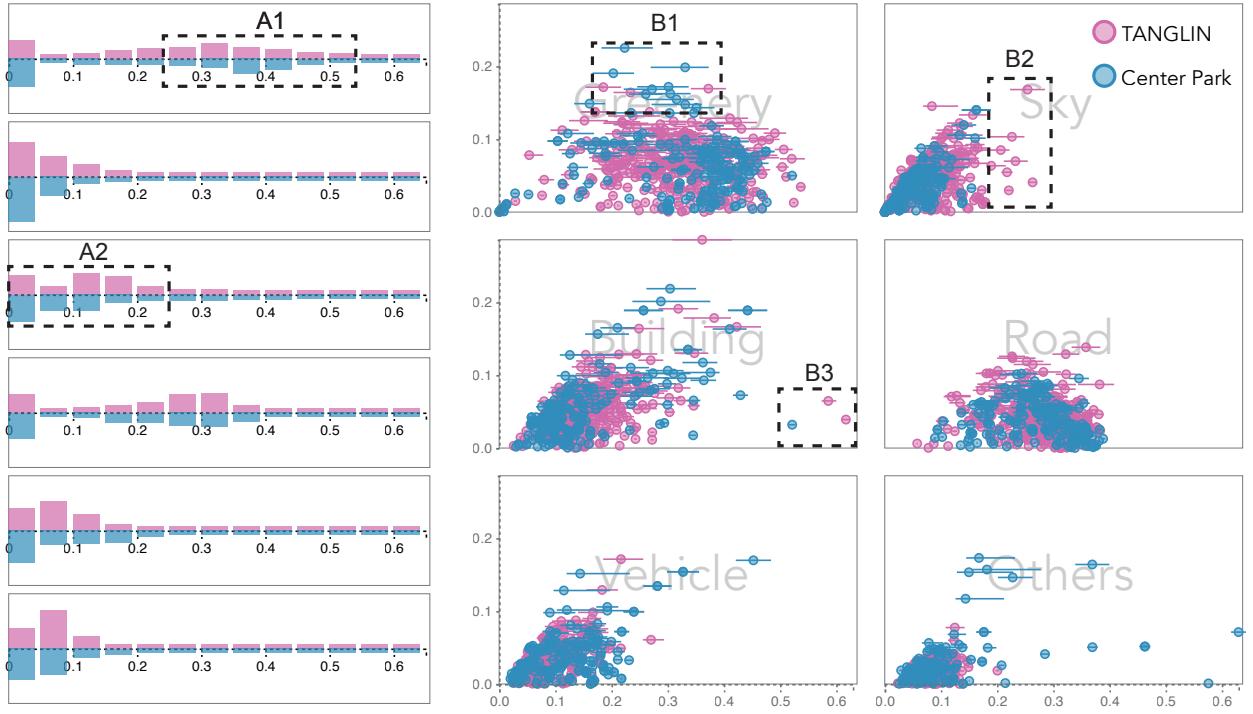


Figure 2.11. Region-scale comparison of Tanglin in Singapore with Central Park in New York City.

high sky visibility in Tanglin (B2), and some streets have very high *building* ratios in both regions (B3).

The results of this study show that human-scale urban forms are correlated with regional functionalities. Meanwhile, even though two regions may have the same functionalities, street spaces in regions can be different between two cities. This may reflect the differences in city planning and development strategies between cities.

### 2.7.3 Street-scale Comparison

Our collaborating domain expert *SR* is interested in comparing the fine-grained details of two streets (T.1.2). StreetVizor meets this requirement with Street Explorer. Here, we select one street from Brooklyn, New York City, and one from Kowloon, Hong Kong. Both streets are representative streets of each district.

Fig. 2.12 presents the visual comparison of these two streets. The map views provide an overview that nearly half of the street views are mostly green (*greenery*), and the other half are mostly orange (*building*) on the street from Brooklyn. By contrast, the street views in Kowloon are

mostly orange (*building*). We can observe more details by looking at the street view images. As shown on the top images, more *greenery* and *sky* with low *buildings* are seen in the left two images, whereas the right two images are filled with *building* and *road*. The tree map of the leftmost image shows the street view is well balanced with *greenery*, *road*, *building*, and *vehicle* features. Notice that the street view is also highlighted in the bottom Street Statistic View. More details on the quantitative measurements of street views can be observed in the bottom Street Statistic View. The themeriver plots clearly show differences in feature distributions: street views in Brooklyn are more mixed with balanced *greenery*, *sky*, *building*, *road*, and *vehicle* features, whereas street views in Kowloon are mostly filled with *building* and *road* features. The histogram bars in PCP further confirm this observation: most street views in Kowloon have less than 10% *greenery* and *sky*.

Though it is well known in the field of urban planning that New York City is well integrated with natural features and that Hong Kong has more high-rise buildings, *SR* is excited to see our system can present these differences so intuitively. “Street Explorer can definitely improve our work efficiency,” *SR* commented.

## 2.8 Expert Review

To evaluate the effectiveness of StreetVizor, we conducted expert interviews with two independent domain experts other than our collaborating senior researcher *SR*. One of them focuses on designing livable public space (denoted as *EA*), and the other is an urban ecologist aiming at improving greenery in cities (denoted as *EB*). Hence, the experts are from different backgrounds: *SR* and *EA* in urban planning, while *EB* in ecology. In the interviews, we started with explaining the visual encodings and interface design, and demonstrated to them how our system works. Then, we showed them the case studies, and allowed them to explore the system by themselves for about twenty minutes in the end. In general, both experts agrees that the way we study human-scale urban forms with street view images is a promising direction. Their detailed feedbacks are summarized below.

*Methodology and Approach.* *EA* agreed with *SR* that evidence-based urban design is becoming a trend in urban planning. He commented, “StreetVizor is far more than simply a visualization platform. Rather, it is an excellent combination of machine learning and visualization techniques, together with classical urban design theories in place-making”. *EB* also expressed that “street view

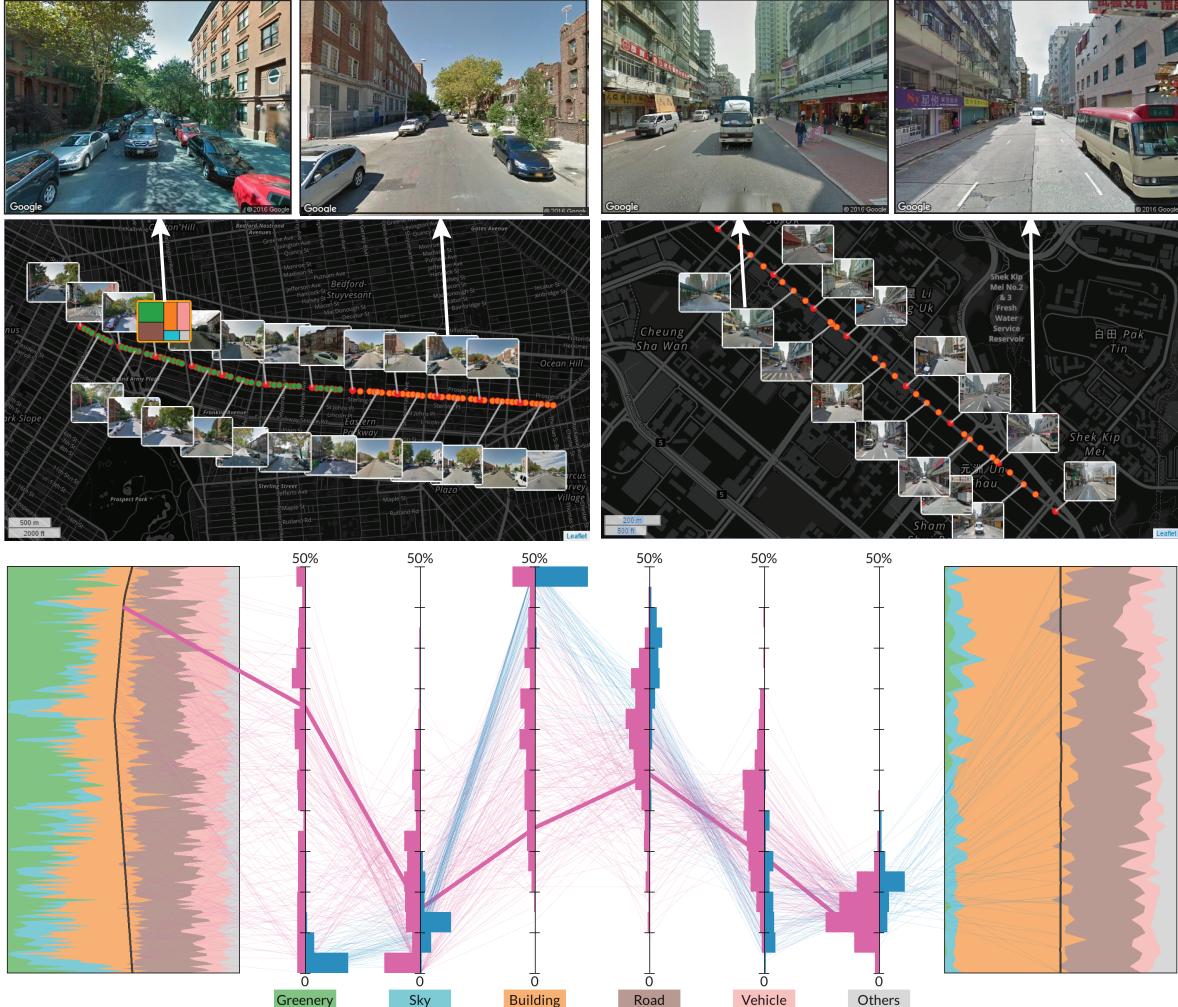


Figure 2.12. Street Explorer compares the differences in human-scale urban forms of two streets in Brooklyn, New York City (left) and Kowloon, Hong Kong (right). The left street views contain more balanced features, whereas the right street views are dominated by *building* and *road*.

images as an emerging data source can reflect urban environments well", and a visual analytics system can greatly facilitate the exploration of street views.

*Interactive Visual Design.* Both experts confirmed that StreetVizor is nicely designed according to the problem domain. They appreciated the visual consistency across different views. *EA* highlighted "it is very important to use the same colors in different views". *SR* agreed the workflow of ranking multiple AOIs/streets with Ranking Explorer, and comparing two AOIs/streets for details through AOI and Street Explorer is helpful. "It is easier for me to identify interested regions, such as those in Hong Kong (Fig. ??)", commented by *SR*. The AOI Explorer is nicely designed with intuitive map and statistic views. In particular, "the statistic view seamlessly integrates three eas-

ily understandable plots”, commented by *EB*. By referring to the scatterplot matrix in Fig. 2.5 & Fig. 2.9, *EA* was excited to see the negative correlation between *greenery* and *building* – “there are much space to improve”. *EA* also liked the visual comparison of human-scale urban form distributions over space (Fig. 2.8), as it reflects the differences of urbanization process and master plans in different cities.

The experts thought presenting street view images in the Street Explorer is intuitive, and mouse hover over to show tree map is helpful. In contrary, it was difficult in the beginning for both experts to understand the PCP enhanced with street layout information, especially the themeriver plot. But after exploration, they agreed it is an excellent idea, as it clearly reveals the feature distributions along street. “Though not common, I believe there are many applications and potentials for the enhanced PCP”, commented by *EB*.

*Applicability.* Both domain experts would like to apply our system to deal with practical problems in their domains. Expertized in urban planning and design, *EA* emphasized “the lack of efficient tools for human-scale management and design obstacles creating high-quality urban streets.” StreetVizor has a great potential to be employed by planners and designers to “build more pedestrian-oriented and livable streets.” *EA* also commented “StreetVizor is highly applicable for evaluating case studies in urban planning”. For example, planners can select several key areas, e.g., CBDs, among different cities and then compare their spatial features to develop appropriate strategies in urban renewal. *EB* would like to apply our system in environment monitoring, since “the large amount of GSV images can provide rich information of urban environment”. He suggested to extend our system in exploring a time-series street view dataset, so that it would allow him to monitor environment changes.

*Limitations and Improvements.* The experts pointed out some limitations in our system. In this work, we explore only six features of human-scale urban forms. Both experts suggested to extract more urban forms, such as aesthetic amenities and mental well-being, from street view images. This will advance our system’s analytical capabilities and extend its applicabilities. For instance, a recent study [58] shows that certain relationship may exist between street greenery & sky ratio and the risk of health challenges. Besides, they proposed to improve our visual designs. *EB* noticed the feature histogram bars in AOI and Street Statistic View are designed differently: one in horizontal, and the other one in vertical style. He felt the vertical histogram bars are confusing, as they “do not

fit our work habits”. *EA* suggested the Street Statistic View can be further improved, by “encoding neighboring street layouts in the plot”, to reflect spatial information better.

## 2.9 Discussion

In this work, we combine automatic machine learning and interactive visual analytics techniques to explore human-scale urban forms. The combination of methods tackles the challenges of integrating information from multiple perspectives and at different scales for analysis. This approach is attractive for urban planners [78, 79] because it shows the possibility of transferring traditional subjective and intuitive-oriented urban design to evidence-based and big-data informed methods.

The analysis of human-scale urban form in this work relies heavily on a deep learning technique for image classification. Therefore, classification accuracy poses serious challenges in our approach. We select SegNet, which achieves a global accuracy of 82.8%, out of the different classification techniques that we tried. The case studies show that the classification technique could provide reasonable analytical results on city- and region-scale human-scale urban form patterns. Nonetheless, the results remain unsatisfactory in many cases, especially when users would like to examine fine detailed urban forms at street-scale. We envision applying a more advanced classification algorithm in the near future, given the rapid evolution of image classification techniques. In addition, the classification is preprocessed offline. Our system does not support the analysis of street views queried on runtime. Thus, its applicability is limited and domain knowledge of planners are underutilized in exploring street views in other cities. This deficiency can also be resolved with advancement in image classification algorithms and machine computing capabilities.

Moreover, we expect that our system will face scalability issues when the number of street view images increase (e.g. when analyzing street views in more cities). To tackle this problem, we can integrate more advanced data structures, such as nanocubes [74] or Gaussian cubes [122] for spatial data querying. More levels of detail and abstraction can also be introduced to handle this problem. The increase in image number will also burden street view clustering in the interactive exploration process. We anticipate that certain pre-configurations will facilitate this process.

Presenting multivariate data with spatial information is challenging. We tackle this challenge by integrating popular PCP with a themeriver plot along an adjusted street layout. The case studies and feedbacks from experts demonstrate the effectiveness of this design. Nonetheless, there are some

issues with our design. First, it represents street layout as a simple spline, which is not sufficiently intuitive when the street is straight. We plan to encode more semantic labels, such as neighboring streets, in the design, as suggested by EA. Second, although the majority of the streets (over 95%) that we have explored can be rotated and fitted in the rendering space, adjustment does not work in some cases. Typical examples are streets in a spiral layout. A more general approach should be developed that can reveal the street layout intuitively and seamlessly fit the street layout in the rendering space.

## 2.10 Conclusion and Future Work

In this chapter, we introduce StreetVizor, a visual analytics system for the exploration of human-scale urban forms based on GSV images. Through discussions with a collaborating researcher who specializes in evidence-based urban planning, we identify various analysis criteria and formulate a set of analytical tasks. We integrate some well-estimated visualization techniques, such as juxtaposition map views, scatterplot matrix, and small multiples, into our system. Specifically, we design an enhanced parallel coordinates with street layout to present coordinated feature values and reveal feature distributions along a street layout. StreetVizor is used to analyze  $\sim 1.7$  million street views from Hong Kong, Singapore, Greater London, and New York City. Using our system, domain experts detect some interesting patterns, such as the negative correlation between *greenery* and *building* features. The experts also agree that our system has a wide range of applications, in areas like public space design and street environment monitoring.

There are several promising directions for future work. First, we would like to extract more high-level information, such as signboards, from GSV images. The signboards can then be compared with points-of-interest information extracted from other data sources, e.g., Foursquare or Google Places. We anticipate revealing some interesting patterns by fusing multiple types of big urban data. In addition, to address the scalability issues, we plan to develop more advanced data structures (e.g., [74, 122]) to improve the querying and filtering processes in our method. Besides, we would like to explore new visual interfaces to compare detailed statistics of human-scale urban forms across multiple AOIs/streets at the same time.

# CHAPTER 3

## ROUTE-AWARE EDGE BUNDLING FOR VISUALIZING ORIGIN-DESTINATION TRAILS IN URBAN TRAFFIC

The visualizations for the large amounts of Origin-destination(OD) trails always cause serious clutter issues, which can be mitigated by edge bundling techniques. In this chapter, we first identify inconsiderate settings of conventional kernel density estimation edge bundling (KDEEB) when applied to urban traffic data, including non-optimal kernel size and road neglect. Then we present route aware edge bundling (RAEB) specifically designed for the OD trails bundling, which addresses these limitations by introducing a comprehensive pipeline comprising preprocessing, bundling and evaluation processes. A series of new parameters, together with adaptions of existing ones, are employed in the pipeline.

### 3.1 Introduction

Movement can be defined as change of an object’s positions or geometric attributes over time [27]. Within a specific time period, the movement of an object can be modeled with an origin (O), a destination (D), and consecutive positions (trail) in-between. Due to fast development of location sensing technologies, massive amounts of OD trails, such as vessel movements and taxi trips, have been collected. Studies of OD trails have revealed many movement patterns and contributed to many applications, e.g. diseases spread control [12] and transportation planning [121].

Visualizing OD trails is a hot yet challenging topic. Conventional method that connects origins to destinations with lines [111] can easily cause visual clutter. To tackle the problem, methods of aggregating trails into flows are usually employed. A number of automatic techniques have emerged, such as intelligent routing layout [94, 116] and spatial mapping [5, 45]. These methods are preferable for revealing overall traffic patterns for answering general questions, e.g., what are popular roads in a city? where do people head to? However, the methods mostly ignores infor-

mation of individual OD trail. In certain scenarios such as to find abnormal people movements, additional analytics are required to complement the visualization.

Wrapping up OD trails into bundles can reveal overall traffic patterns, meanwhile keep individual OD trail. Many bundling methods including spring-based (e.g., [52, 100]), image-based (e.g., [56, 70]), and geometry-based (e.g., [24, 51]), have been proposed. These techniques have been successfully employed for many different datasets such as Internet connections and migrations. In these data, it is flexible to manipulate edges as long as node connections are kept. This, however, does not hold for OD trails in urban traffic data. In a city, humans and vehicles are moving along roads, and many activities including traffic jams and accidents, are happening on roads. Hence, OD trails are constrained by road networks. Trail information is equally, if not more, important as ODs for urban traffic data.

Among various edge bundling methods, kernel density estimation (KDEEB) [56] is in state-of-the-art with advantages in speed and generality. However, we identify several inconsiderate settings of KDEEB when applied to OD trails in urban traffic data, including non-optimal kernel size and road neglect. These settings wreck applicability of KDEEB for movement visualization, which requires to preserve road network topology and to support multi-scale exploration. This work presents a new method, namely route-aware edge bundling (RAEB), specifically designed to address the limitations. RAEB employs a comprehensive pipeline consisting of preprocessing, bundling and evaluation processes. A series of new parameters such as route awareness and bundling stability, which leverage advanced techniques from geography and image processing fields, are introduced. Experiments on artificial and real-world urban traffic data show that RAEB can generate more realistic results, meanwhile achieve fast computation speed comparable to KDEEB.

The primary contributions of this work include:

- We develop a new edge bundling method of RAEB, which includes a comprehensive pipeline consisting of preprocessing, bundling, and evaluation processes.
- We adapt existing parameters in KDEEB, and introduce a series of new parameters, to better fit OD trails in urban traffic.
- We conduct several experiments using artificial and real-world urban traffic data to demonstrate effectiveness of RAEB.

## 3.2 Related Work

**Urban Traffic Visual Analytics.** In order to facilitate exploration of human/vehicle/goods movement in cities, many visual analytics for urban traffic data have been developed. These visual analytics can facilitate understanding of urban dynamics and human activities, and also enhance traffic management and assessment. Systematic reviews of these works are presented in [4, 18]. In particular, some visual analytics have been specifically developed for studying OD trails in urban traffic. Among them, advanced indexing methods (e.g., [33, 130]) and new interaction models (e.g., [62, 130]), have been developed to facilitate movement query. After filtering OD trails, these visual analytics are typically complemented with statistical analysis, and the analysis results are then presented with statistical graphics.

Figure 3.1(a) presents a density map of Manhattan green taxi trips mapped on road network. The visualization method is employed in many visual analytics for urban traffic. Though commonly available, the visualization requires additional analytics to reveal overall traffic patterns. This work aims to develop a new visualization technique which can present traffic patterns in one view. Such a visualization can be complementary with existing analytics techniques.

**OD Movement Visualization.** Urban traffic such as taxi trips and mobile phone traces can be categorized as movement data. The data are commonly modeled as OD trails containing information of origin, destination, and in-between trails. Many OD visualization techniques have been devised, which in general can be categorized into matrix- and flow-based. Matrix-based techniques represent ODs as a matrix, with cell colors representing flow magnitudes of OD pairs. Sorting of rows and columns are usually used to make cluster patterns apparent [124]. The techniques can clearly show OD flow magnitudes, yet it fails to present geographical information. OD map [125] was devised to address the shortage. The approach divides a traditional OD matrix into two layers, where the first and second layer represents origins and destinations, respectively. Flow-based techniques provide a more intuitive way to show geographical information. However, the methods face scalability issues for huge amounts of OD movements, as the display will be cluttered with excessive edge crossings. To mitigate clutters, spatial aggregation methods are usually employed. OD aggregation methods such as graph partitioning [44] and density-based clustering [118], have been proposed to cluster ODs. Spatial aggregation can also be accomplished by grouping trails into continuous flows in flow map layout [94] or composite layout [22].

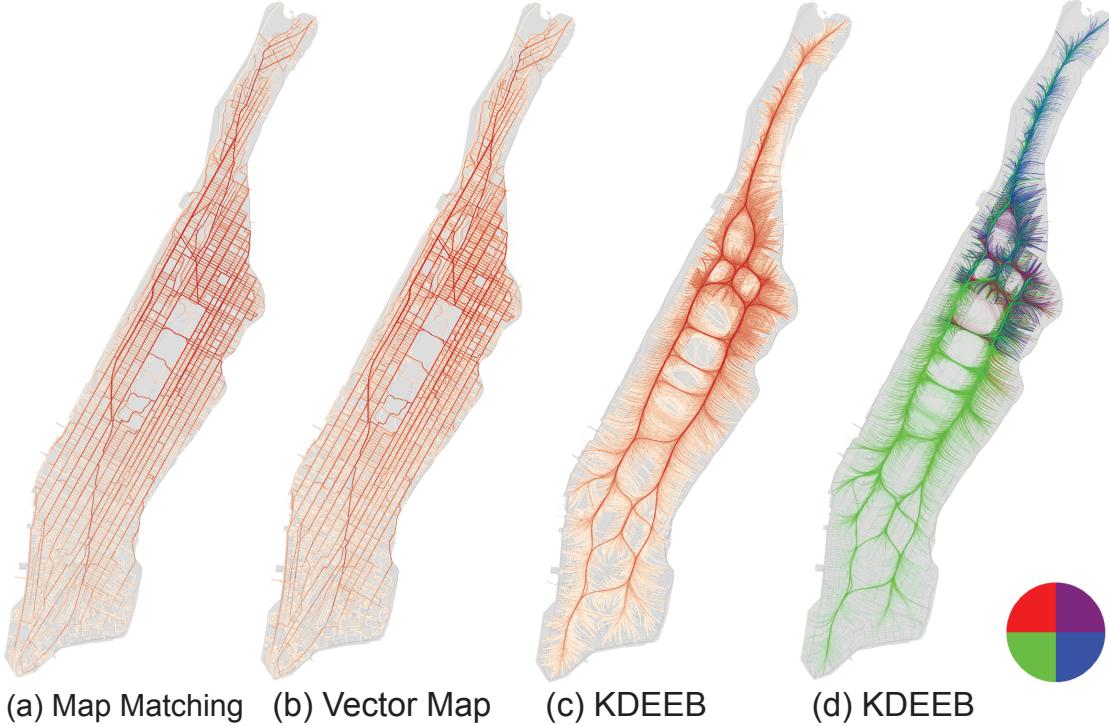


Figure 3.1. Methods for visualizing OD trails in urban traffic: (a) map matching, (b) vector map, (c) & (d) KDEEB bundles rendered in density and direction, respectively.

Both matrix- and flow-based methods perform well in reducing clutter, yet information of individual OD trail is not preserved. This may cause unexpected information loss when exploring urban traffic data. For example, an abnormal suspect movement to a remote place can be grouped together with normal trips.

**Edge Bundling.** To show individual OD trail, we treat ODs as nodes, and trails as links in a node-link graph, and employ edge bundling methods to reduce clutters. Systematic overviews of state-of-the-art edge bundling techniques can be found in [69, 136]. Here, we just give an overview.

Methods for general graphs can be categorized in three categories [136]. *Spring-based* methods model edges as springs, and generate bundled graph layout by assigning forces to the edges that can attract or repel each other (e.g., [52]). These methods usually suffer from high computational complexity and the processes are time-consuming. *Geometry-based* methods cluster edges based on potential control meshes in the input graph, which can be constructed through hierarchical graph drawing [51], triangulation [67, 137], complex polygons [24], and functional decomposition [55]. Control meshes make geometry-based methods flexible and easy to control, yet the construction process can be time-consuming when the graph is large. *Image-based* methods utilize image-

processing to accelerate the bundling process. Skeleton-based (SBEB [30]) and kernel density estimation (KDEEB [56]) methods are developed in this category. Compared with the other two groups of methods, image-based methods leverage GPU’s powerful computing capacity, making them computationally efficient and scalable to handle millions of edges [70, 113].

Besides, many edge bundling methods are developed targeting at improving certain features such as to minimize ambiguity [9, 83] and to reveal directional edge patterns [101]. Some other methods are developed for specific applications such as to visualize geographical data in 3D [66, 110] and neural connections in brain [13, 129]. Closely related to our work, vector map [110] employs road network as a reference graph for edge bundling and represent the bundles as B-splines. The approach is preferable for avoiding intersecting with obstacles in 3D, while marginal improvements are accomplished for urban traffic data on dense road networks. Figure 3.1(b) presents a vector map for Manhattan taxi trips in finest scale, which is very close to the original map as in Figure 3.1(a). In comparison, Figure 3.1(c) & (d) present KDEEB bundles with bundle lines colored in flow density and OD direction, respectively. Main traffic roads and flow directions can be identified easily. Nevertheless, there are many limitations of KDEEB when applied to urban traffic data. We address these limitations with a new bundling method of RAEB.

### 3.3 Problem Statement and System Overview

This section briefly describes KDEEB algorithm, followed by a discussion of inconsiderate settings when applied to OD trails in urban traffic. An overview of RAEB is presented at the end.

#### 3.3.1 KDEEB Algorithm

KDEEB is a fast edge bundling method for visualizing dense graphs. It employs a general pipeline with edge sampling, splatting, gradient estimation, edge advection, and bundle smoothing. In edge sampling step, an edge  $\mathbf{e}_i$  is divided into uniformly-sampled polylines, i.e.,  $\mathbf{e}_i = \{\mathbf{x}_j\}$ , with user-given sampling step  $\sigma$ . After that, KDEEB computes a density map  $\rho : \mathbb{R}^2 \rightarrow \mathbb{R}^+$  as

$$\rho(\mathbf{x} \in \mathbb{R}^2) = \sum_{\mathbf{y} \in D} K\left(\frac{\|\mathbf{x} - \mathbf{y}\|}{p_r}\right) \quad (3.1)$$

where  $K$  is a parabolic radial kernel and  $p_r$  is kernel radius. Next, each edge sampling site  $\mathbf{x}_j$  is

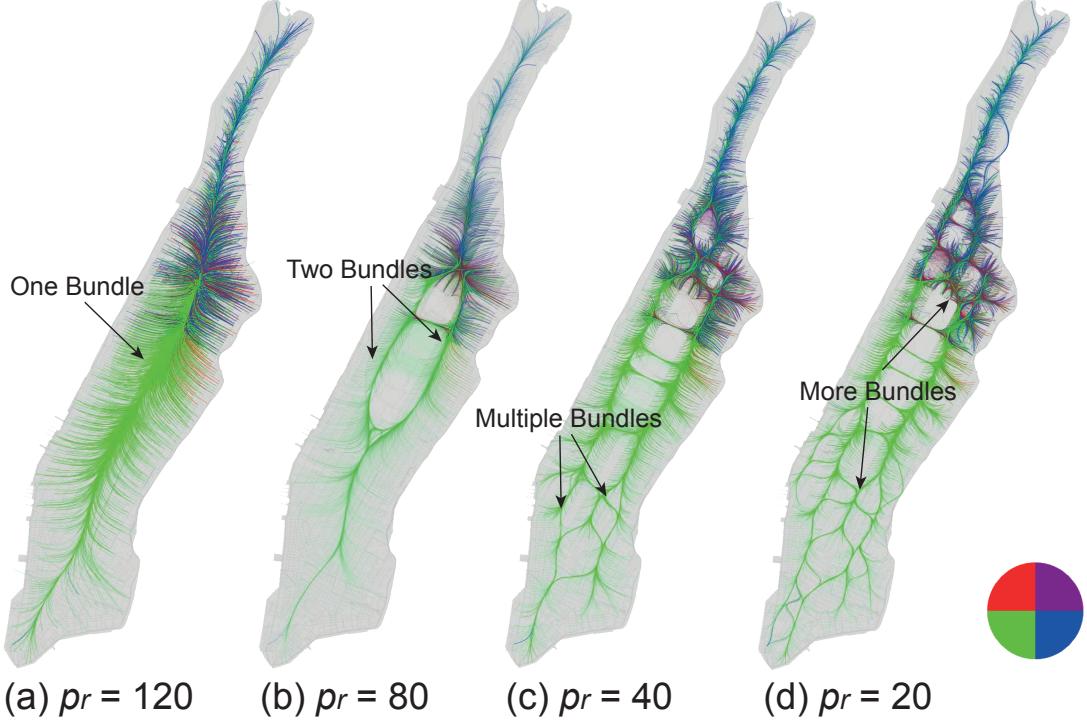


Figure 3.2. KDEEB applied to taxi trips in Manhattan with different kernel sizes  $p_r$ : (a) 120, (b) 80, (c) 40, and (d) 20. More detailed bundles are generated with smaller  $p_r$ .

adverted to a new position  $\mathbf{x}'_j$  as:

$$\mathbf{x}'_j = \mathbf{x}_j + p_r \frac{\nabla \rho}{\|\nabla \rho\|} \quad (3.2)$$

Finally, a smoothing function is applied on  $\mathbf{x}_j$  to remove jitters created by the advection step. These steps are repeated  $p_n$  times until stable bundles are generated.

*Parameter Setting.* KDEEB requires several parameters to be preset by users. Following settings are recommended [113]:

- Kernel radius  $p_r$ : Initial value set to 5% of graph drawing size. A constant decay ratio  $\lambda$  in  $[0.5, 0.9]$  is applied to  $p_r$  at each bundling iteration  $n$ .
- Bundling iteration  $p_n$ : An integer in between 10 to 15.

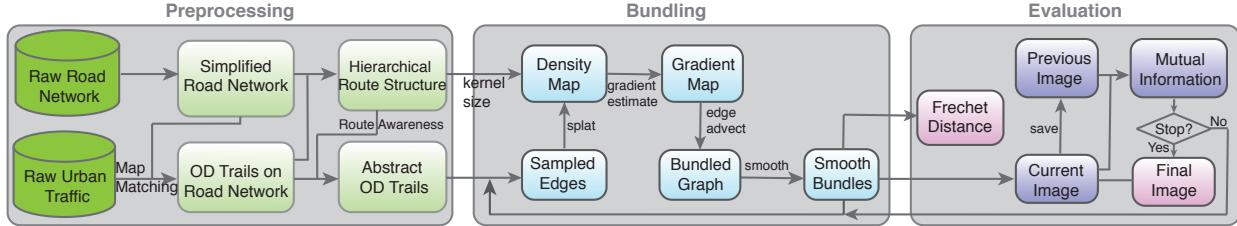


Figure 3.3. Overview of RAEB pipeline. Our method mainly consists of three phases: *Preprocessing* for creating a proper initial layout, *Bundling* for bundling input OD trails, and *Evaluation* for generating a stable bundle structure.

### 3.3.2 Problem Identification

The parameter settings are recommended for general graphs that consist of fixed nodes and connections between them. For such a graph, it is free to manipulate node connections into smooth polylines. However, when applied to OD trails in urban traffic data, KDEEB encounters the following issues.

- *P1: Non-optimal Kernel Radius.* Kernel radius  $p_r$  in KDEEB is solely based on graph drawing size, while ignores geometric properties of road network. Figure 3.2 presents KDEEB results generated with different kernel sizes for taxi trips in Manhattan: 120, 80, 40, and 20 in (a) - (d), respectively. Sizes for graph drawing are all  $720 \times 1440$ , thus KDEEB would recommend  $p_r = 80$  as 5% of graph graphing size. However, smaller  $p_r$  generate more detailed and visual appealing bundles, comparing Figure 3.2(b) to (c) & (d). This happens because most space in graph drawing are not utilized, due to the long and thin shape of Manhattan island. In addition, urban traffic can unevenly spread in a city, such as Shenzhen taxi data (Section 3.7.3), making small  $p_r$  preferable.
- *P2: Road Neglect.* Urban traffic are moving along roads in a city, thus OD trails should be constrained to roads in a city. However, KDEEB represents each OD connection with a direct line, while ignores information about the road network. Thus, resulting bundles can be displaced far away from the roads. This can cause impression of traffic moving in wrong positions, such as those bundles in between Manhattan and Brooklyn (Section 3.7.1). The displacements can also add difficulty to mentally map original OD trails with resulting bundle lines.

Besides, KDEEB lacks a quantitative method to automatically terminate bundling iterations.

- *P3: Preset Bundling Iterations.* KDEEB is an image-based edge bundling method. Most of these bundling methods generate bundling results in an iterative way. The number of bundling iterations  $p_n$  is preset based on practical experience: 10 in KDEEB [56], while 15 in CUBu [113]. The numbers are chosen because tight and stable bundles are generated. However, these conditions are rather subjective, which can vary among individuals and may be affected by other parameters like image size and sampling step.

### 3.3.3 RAEB Overview

We develop route-aware edge bundling (RAEB), a comprehensive edge bundling method specifically designed for visualizing OD trails in urban traffic. As illustrated in Figure 3.3, RAEB pipeline mainly consists of three parts: *Preprocessing*, *Bundling*, and *Evaluation*.

In *Preprocessing*, RAEB first generates a simplified road network from the input one such as open street map (OSM). Raw OD trails are mapped onto simplified road network using map matching algorithms. Together with urban traffic information, simplified road network can be organized in a hierarchical route structure. OD trails can be abstracted accordingly depending on a new parameter of route awareness defined by users. A value for kernel size is also measured based on geometric property of the route structure.

The abstract OD trails are passed into *Bundling* stage as input graph. Bundling process is an image-based approach similar to KDEEB workflow. Specifically, we discard the preset parameter of bundling iteration  $p_n$  in KDEEB. Instead, we introduce a new parameter of bundling stability, which is measured as normalized mutual information between images generated by two consecutive iterations. The measurement is done in the *Evaluation* stage. If bundling stability reaches a threshold, we terminate the bundling iteration and export a final image. Besides, we measure Fréchet distance between generated bundles to OD trails mapped on the road network. The measurement complements final image with a quantitative metric that reveals quality of a bundling method.

## 3.4 Preprocessing

This work leverages both OD trails and road network to improve edge bundling quality. This section introduces major steps in *Preprocessing* stage, including: map matching, hierarchical road

structure construction, and trail abstraction.

### 3.4.1 Basic Traffic Concepts

*Road Network:* A road network can be modeled as a directed graph  $\bar{G} = (\bar{V}, \bar{E})$ , where  $\bar{V}$  is the set of nodes in  $\bar{G}$  and  $\bar{E}$  is the set of edges connecting nodes in  $\bar{V}$ . Degree of a vertex  $\bar{v}$ , denoted as  $\deg(\bar{v})$ , indicating the number of edges connected to  $\bar{v}$ . By this, a vertex  $\bar{v} \in \bar{V}$  can be classified as an endpoint with  $\deg(\bar{v}) = 1$ , a midpoint with  $\deg(\bar{v}) = 2$ , or an intersection with  $\deg(\bar{v}) > 2$ . To facilitate computation, we can simply  $\bar{G}$  into a simplified graph  $G = (V, R)$ , where  $V$  is the union of endpoints and intersections, and  $R$  is a set of links connecting nodes in  $V$ . Each link  $r$  represents a sequence of  $n$  consecutive nodes in  $\bar{V}$ :

$$r := \bar{v}_1 \rightarrow \bar{v}_2 \rightarrow \dots \rightarrow \bar{v}_n, \quad \forall \bar{v}_i \in \bar{V} \quad (3.3)$$

where  $\bar{v}_1$  and  $\bar{v}_n$  are endpoints or intersections, while the other nodes are midpoints. For the sake of clarity, we use the term *route* to describe such a link. Computation tasks, including shortest path query and map matching, are performed on the simplified graph  $G$ .

*Urban Traffic:* Thanks to open data campaign, many urban traffic data are available nowadays. Typically, raw urban traffic data can come in either of the following forms:

- *UT-1:* origin and destination locations only, such as New York taxi trips [33] and Singapore public transportation rides [130]. Such a raw trip  $t_{raw}$  can be represented as  $t_{raw} := l_o \rightarrow l_d$ , where  $l_o$  and  $l_d$  represent origin and destination locations, respectively.
- *UT-2:* a sequence of GPS positions, such as Stuttgart scooter fleet [62] and Hangzhou taxi trips [119]. Such a raw trip  $t_{raw}$  can be represented as  $t_{raw} := l_1 \rightarrow l_2 \rightarrow \dots \rightarrow l_n$ , where  $l_i \subset \mathbb{R}^2$ .  $l_1$  and  $l_n$  are origin and destination, i.e.,  $l_o$  and  $l_d$ .

Both forms of urban traffic data can be generalized as a traffic graph  $G_t = (L, T)$ , where  $L$  is the set of locations and  $T$  is the set of trips represented as sequential connections of locations.

### 3.4.2 Map Matching

Constraining movements over an underlying road network, i.e., mapping traffic graph  $G_t := (L, T)$  to road network  $G := (V, R)$ , is usually a preliminary step for urban traffic analysis. This is neces-

Category	Exemplary OSM Indicator	Score ( $r_{hier}$ )
Expressway	motorway, trunk	1
Trunk Road	primary, motorway_link	0.75
Secondary Road	secondary, tertiary	0.5
Branch Road	unclassified, residential	0.25

Table 3.1. Route hierarchy, OSM indicator and hierarchy score  
sary. First, recorded locations get errors due to imprecise GPS localization, while road network is precise and constant. Second,  $G_t$  can be much more complex than  $G$ : Taking New York traffic for an example, there are over millions of taxi trips per day, resulting in millions of locations  $L$ , yet its simplified road network contains less than 100K routes. Therefore, map matching can enrich effective analysis and visualization.

There are many map matching methods available, including shortest path, minimum turns, and ST-matching [81] for GPS traces. These methods are preferable for different dataset. We employ the following methods for corresponding urban traffic.

- For *UT-1*, i.e.,  $t_{raw} = l_o \rightarrow l_d$ , we first find corresponding closest nodes  $v_o \in V$ ,  $v_d \in V$  for  $l_o$  and  $l_d$  respectively. After that, we search for the shortest path between  $v_o$  and  $v_d$  in  $G$ .
- For *UT-2*, i.e.,  $t_{raw} = l_1 \rightarrow l_2 \rightarrow \dots \rightarrow l_n$ , we use ST-matching algorithm, which considers not only spatial features of road network, but also temporal constraints. The method was later visually investigated and optimized with an interactive interface [61].

Both methods return a sequence of routes  $r_1 \rightarrow r_2 \rightarrow \dots \rightarrow r_n$  in  $G$  for each OD trail  $t_{raw}$ . We then combine the routes with origin ( $l_o$ ) and destination ( $l_d$ ) locations of  $t_{raw}$ . Thus, both types of  $t_{raw}$  are converted to the same map matching form.

$$t_{map} := l_o \rightarrow r_1 \rightarrow r_2 \rightarrow \dots \rightarrow r_n \rightarrow l_d. \quad (3.4)$$

Here the OD trail's origin  $l_o$  and destination  $l_d$  are kept.

### 3.4.3 Hierarchical Route Structure Construction

A road network is usually designed in hierarchical levels to avoid dense direct connections among regions. This inspires us to construct a hierarchical route structure to benefit edge bundling. Al-

though urban roads are organized hierarchically, it does not consider OD trails. This work makes use of the following attributes of a route  $r$  in  $G := (V, R)$  to construct a hierarchical route structure.

*Route length ( $\text{len}(r)$ )*. Visual appearance of each OD trail is determined by its routes on road network. Longer a route, more it will affect the visual appearance. Thus an important component affecting route hierarchy is route length. Here,  $\text{len}(r)$  is measured in Euclidean space and normalized over the whole road network.

*Road hierarchy ( $\text{hier}(r)$ )*. Urban roads are classified into hierarchies: freeways, arterials, collectors, and local roads. Similar descriptions can also be found elsewhere [119]. A higher level usually indicates faster traffic speed and less access to property. OSM provides more detailed indicators of road hierarchy, e.g., trunk, primary, tertiary, etc. To keep consistent, we classify these indicators into four hierarchies. A route  $r$  in each hierarchy is assigned an score according to Table 3.1.

*Flow magnitude ( $\text{flow}(r)$ )*. Flow magnitude measures how many OD trails pass through a route  $r$ . The previous two factors purely rely on road network, while  $\text{flow}(r)$  reflects the importance of a route in consideration of OD trails. Here,  $\text{flow}(r)$  is measured by counting how many mapped OD trails passing through  $r$ , and then normalized.

By this, each route can be represented by a vector  $V(r) := \langle \text{len}(r), \text{hier}(r), \text{flow}(r) \rangle$ . Score of a route  $\text{score}(r)$  can be measured as  $\text{score}(r) = \sum w_i \times V(r)_i$ , where the weights  $w_i (i = 1, 2, 3)$  controls how much each attribute affects route weight. We choose 0.3, 0.1, and 0.6 in this work, which highlights the importance of flow magnitude. In the end, all routes are sorted in descending order according to their weights. The routes  $R$  are further subdivided into multiple hierarchical levels  $R^1, R^2, \dots, R^n$ . Each level of routes accounts for top-k routes.

### 3.4.4 Trail Abstraction

**P2** states that road network is omitted in original KDEEB, which can lead to wrong depiction in geographical space and poor support of multi-scale exploration. We introduce a new parameter of route awareness  $p_{ra}$ , which defines the level of trail abstraction, to tackle this issue. Higher  $p_{ra}$  indicates that more details of OD trails should be preserved. Given a hierarchical route structure in  $n$  levels,  $p_{ra}$  can range from 0 to  $n$ . If  $p_{ra}$  is set to zero, OD trails will be abstracted to straight lines connecting ODs directly. If  $p_{ra}$  is set to  $n$ , no abstraction will be applied. If  $p_{ra}$  is set to a value  $m$

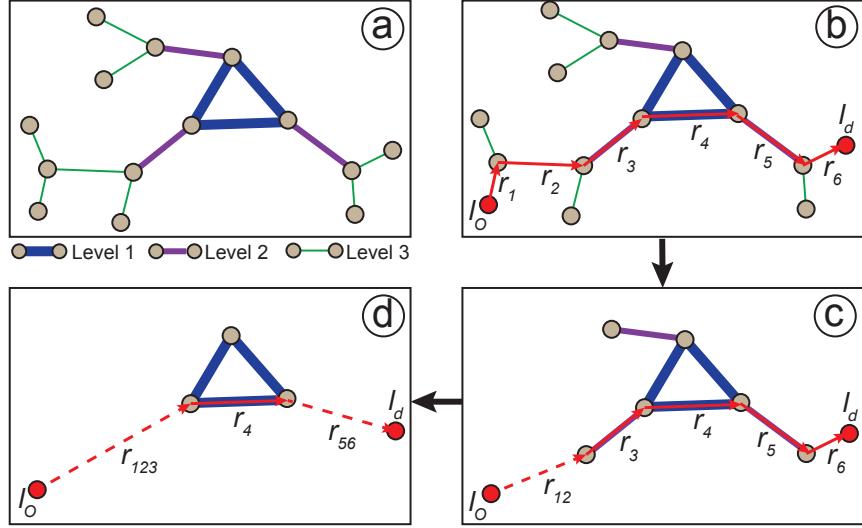


Figure 3.4. Illustration of hierarchical route structure and OD trail abstraction: (a) Three route levels are constructed colored in blue, purple and green, respectively; a raw OD trail is abstracted in accordance to route (b) level-3, (c) level-2, and (d) level-1.

in  $[1, n - 1]$ , routes in level-1 to level- $m$  will be preserved.

Figure 3.4(a) presents an exemplary road network with a hierarchical route structure. The routes are categorized into three levels: blue thick lines for level 1, purple medium lines for level 2, and green thin lines for level 3. Figure 3.4(b-d) illustrates an example of how a raw OD trail of  $l_o \rightarrow r_1 \rightarrow \dots \rightarrow r_6 \rightarrow l_d$  can be abstracted in accordance to the hierarchical route structure shown in Figure 3.4(a). In Figure 3.4(b),  $p_{ra}$  is set 3, and all routes are preserved. In Figure 3.4(c),  $p_{ra}$  is set 2, so route  $r_1$  and  $r_2$  will be merged together as they are both level-3 routes. In Figure 3.4(d),  $p_{ra}$  is set 1, thus only route  $r_4$  will be kept.

## 3.5 Bundling Method

RAEB adapts KDEEB bundling algorithm [56] to better reveal topology of urban traffic. This section presents amendments employed by RAEB, including kernel size measurement and density map generation.

### 3.5.1 Optimal Kernel Size

KDEEB only considers the size of graph drawing when determining kernel size. This strategy may not be optimal for bundling OD trails in urban traffic ( $PI$ ). Ideally, the chosen kernel size should be

---

**Algorithm 1** KernelSizeMeasurement

---

**Input:** Top  $N$  sorted routes as polyline list  $\mathbf{P} = \{P_1, \dots, P_N\}$   
**Output:** Initial kernel size  $p_r$

- 1: Let  $d[ ][ ]$  denote distance between polyline pairs
- 2: **for**  $i = 1$  to  $N$  **do**
- 3:   **for**  $j = i + 1$  to  $N$  **do**
- 4:      $d[i][j] = d[j][i] = \text{DiscreteFrechetDistance}(P_i, P_j)$
- 5: Let  $\mathbf{C}$  denote a list of polyline clusters;
- 6:  $\mathbf{C} = DBSCAN(\mathbf{P}, \text{eps}, \text{minNum});$
- 7:  $C_{max} = argmax_{|C_i|} (C_i | C_i \in \mathbf{C});$
- 8: Initialize  $d_{geo} = 0;$
- 9: Let  $count = |C_{max}| \times (|C_{max}| - 1);$
- 10: **for each**  $P_i \in C_{max}$  **do**
- 11:   **for each**  $P_j \in C_{max}$   $\&& P_i \neq P_j$  **do**
- 12:      $d_{geo} = d_{geo} + d[i][j] / count;$
- 13:  $p_r = \text{ToDrawingSpace}(d_{geo} / 2);$
- 14: **return**  $p_r$

---

able to bundle closely-related OD trails (e.g., movements on bi-directional roads), while separate loosely-correlated OD trails (e.g., movements on two different highways).

To address  $P1$ , both graph drawing size and geometric property of road network should be considered. We develop an automatic process for measuring initial kernel size, as outlined in Algorithm 1. Here, top  $N$  routes are extracted from *Preprocessing* stage. The routes are treated as a polyline list  $\mathbf{P}$ , and inputted into the algorithm. Pair-wise distance between each pair of ploylines are computed. Then a DBSCAN algorithm is applied to group the polylines into polyline clusters  $\mathbf{C}$ . The cluster with maximum number of polylines is extracted, and denoted as  $C_{max}$ . Average geographical distance  $d_{geo}$  is computed for polylines in  $C_{max}$ . Finally, half of  $dist_{geo}$  is converted to  $p_r$  in graph drawing space, and  $p_r$  is returned as the initial kernel size.

Basically, the algorithm first computes average geographical route distance in the top  $N$  routes, and then converts the distance to graph drawing space.  $N$  should not be too small that the routes are not representative for the whole road network; meanwhile, it should not be too big to overlap route awareness. From our experiments, we find 1% of whole route size is appropriate for  $N$ . Besides, another important factor is the distance metric between two polylines. Here, we choose discrete Frechet distance, which measures similarity of two polylines considering both locations and ordering of the points along the polylines. The metric is one of the most popular methods for movement analysis [43], and can be computed efficiently [29].

### 3.5.2 Density Map Generation

KDEEB omits trail information (*P3*), as it connects ODs with directed lines. We aim to keep bundles close to OD trails on road network. More specifically, in *Preprocessing* stage, each OD trail is abstracted into a sequence of artificial directed lines and real routes in road network. The artificial directed lines are free to manipulated, while we would like to keep the bundle layout spatially close to the real routes. In addition, it is preferable to keep the bundling paradigm of advecting sampling sites towards high density directions.

To achieve this goal, we modify the density map generation used by KDEEB. Let denote a list of routes  $R_{\text{aware}} \subset \mathbb{R}^2$  are kept for route awareness. Since sampling sites are advected towards their gradient directions, gradients of points along  $R_{\text{aware}}$  should be either (1) zero, or (2) pointing to route direction. To minimize effects on densities surrounding  $R_{\text{aware}}$ , we choose the second option. Instead of Eqn. 3.1, we now use

$$\rho_{od} = \rho + D(\mathbf{x} | \mathbf{x} \in R_{\text{aware}}) \quad (3.5)$$

where  $D(\mathbf{x}) \in \mathbb{R}^+$  is a constant assigned to a point  $\mathbf{x} \in R_{\text{aware}}$ . High  $D$  value will make it more likely that gradient at  $\mathbf{x}$  points to the same direction with the route.

## 3.6 Evaluation

Besides introduction of *Preprocessing* and adaptions in *Bundling*, post hoc bundling evaluation methods are also developed in RAEB. This section introduces these methods, including mutual information and bundle deviation measurements.

### 3.6.1 Normalized Mutual Information

After each bundling iteration, a bundling method needs to decide whether the bundling iteration should continue or not. KDEEB presets a bundling iteration parameter  $p_n$ , similar to many other bundling methods including SBEB, FTTEB, and CUBu.  $p_n = 10$  is chosen such that the bundles converge to a stable structure. Structure stability is basically the perception of visual similarity between images generated in two consecutive bundling iterations. However, the perception can be

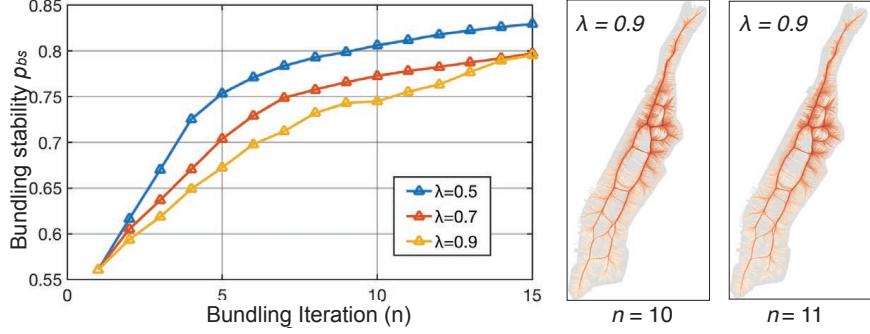


Figure 3.5. (left) Bundling stability  $p_{bs}$  measured at each bundling iteration for different decay ratios  $\lambda$ , (right) visually indistinguishable images are generated at iteration 10 and 11 for  $\lambda = 0.9$ .

easily biased by many factors, including people's color perception difference, lighting conditions, etc. It lacks a quantitative method for measuring structure stability.

We employ mutual information (MI) - a basic concept from information theory, as a quantitative indicator for measuring bundling stability. Maes et al. [84] introduced MI to measure statistic dependence between intensities of corresponding voxels in two medical images. This work measures MI between images generated in consecutive bundling iterations, which are geometrically aligned. Hence, the measurement is simplified without geometry matching. Let denote two input images  $X$  and  $Y$ . MI is computed as:

$$I(X, Y) = \sum_x \sum_y p(x, y) \log\left(\frac{p(x, y)}{p(x)p(y)}\right) \quad (3.6)$$

where  $p(x)$  and  $p(y)$  indicate the intensity at  $x \in X$  and  $y \in Y$ , and  $p(x, y)$  measures the joint value between  $p(x)$  and  $p(y)$ . MI is affected by image size, while we prefer a constant measurement for different graph drawings. Hence, we adopt normalized MI (NMI):

$$NMI(X, Y) = \frac{2I(X, Y)}{H(X) + H(Y)} \quad (3.7)$$

where  $H(X) = -\sum p(x) \log(p(x))$  that measures entropy for  $X$ .

We introduce a new parameter of bundling stability  $p_{bs}$ .  $p_{bs}$  is computed in every iteration as  $NMI(I_n, I_{n-1})$ , where  $I_n$  and  $I_{n-1}$  denote images generated at iteration  $n$  and  $n - 1$ , respectively. Figure 3.5(left) presents NMIs at each bundling iteration for three different kernel size decay ratios  $\lambda$ . It is observed that NMI increases when bundling iteration increases, and smaller  $\lambda$  leads to faster NMI increment. All NMIs pass over 0.75 around iteration 10, and reach close to 0.8 at iteration 15. Figure 3.5(right) shows two images at iteration 10 and 11 for  $\lambda = 0.9$ . The images are hardly

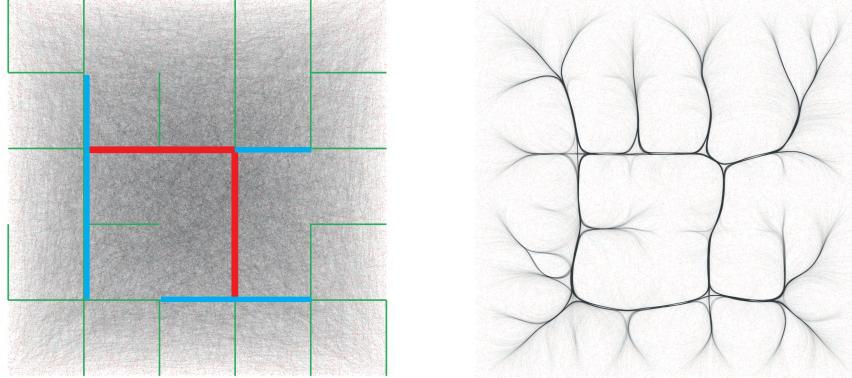


Figure 3.6. Left: raw 100K artificial OD trails on a hierarchical road network. Right: RAEB bundling result with  $p_{ra}$  set to 2.

distinguishable from each other. Thus, we can claim the bundling produces stable results, and the process can terminate. In this way, we determine a threshold for  $p_{bs}$  to control bundle termination. If a bigger threshold is chosen, more stable results will be generated.

### 3.6.2 Bundle Deviation

$p_{bs}$  is introduced for measuring visual stability of bundle layouts generated by one bundling method. However,  $p_{bs}$  is not suitable for comparing bundling quality of multiple bundling methods. Many evaluation metrics can be used for comparing graph layouts, such as edge crossing, ambiguity, and edge direction. These metrics are suitable for general graphs without a ground truth. However, OD trails studied in this work are constrained in roads in a city. We would like to keep the generated bundles close to their OD trails on the road network. Though map matching algorithm can introduce errors, this work treats mapped OD trails as ground truth, instead of raw OD trails. This is because of: for *UT-1* urban traffic, only origin and destination information are recorded; for *UT-2* urban traffic, accuracy of GPS positions are not perfect.

We introduce a new parameter of bundle deviation, denoted as  $p_d$ , to measure distance between final bundles and OD trails mapped on the road network. Both bundles and OD trails can be modeled as polylines made up of sequential positions in geographical space. Thus, the distance between them can be measured using discrete Frechet Distance. We calculate mean distances for all bundles, and map the mean in graph drawing space.

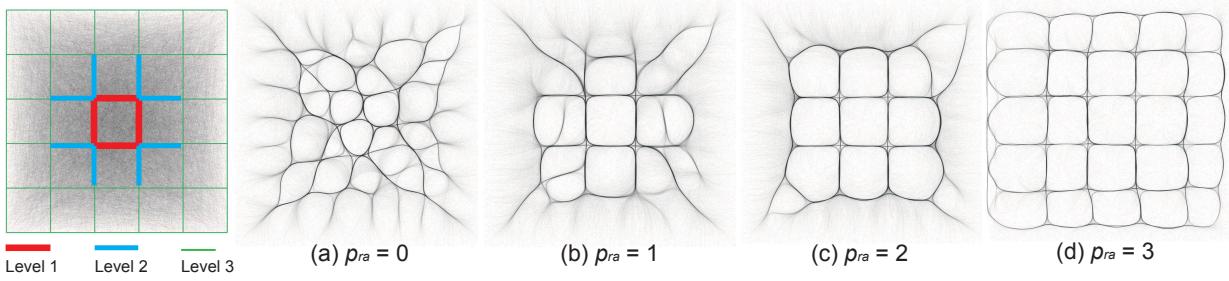


Figure 3.7. Leftmost: raw 100K artificial OD trails on a grid road network in three hierarchies. (a) - (d): RAEB bundling results with route awareness parameter  $p_{ra}$  set to 0 - 3, respectively.

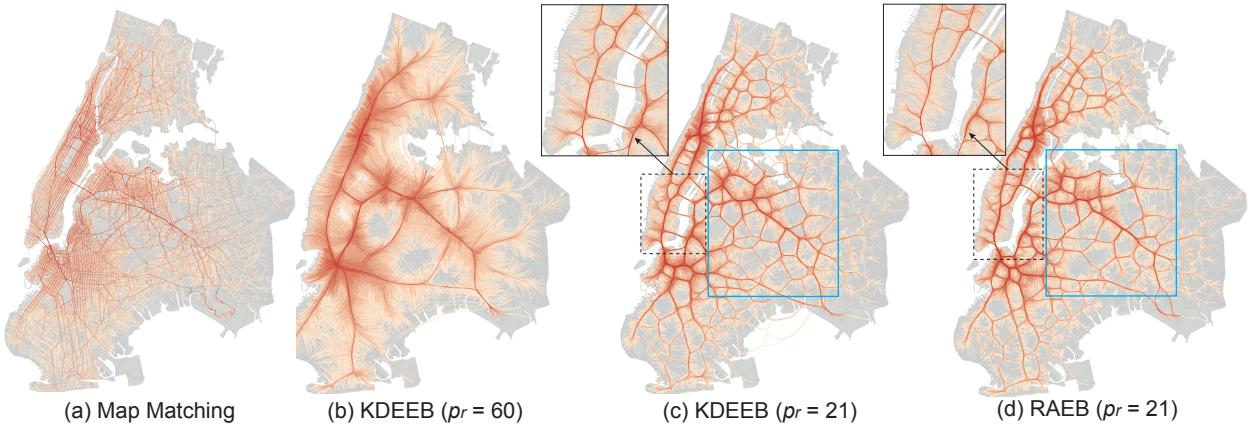


Figure 3.8. Density maps of NYC taxi trips: (a) shortest paths mapped onto the road network, (b) KDEEB bundles with kernel size  $p_r = 60$ , (c) KDEEB bundles with  $p_r = 21$ , and (d) our RAEB bundles with  $p_r = 21$ .

## 3.7 Applications

This section presents applications of RAEB on artificial OD trails, and real-world taxi data in New York and Shenzhen. We show advancements of RAEB over KDEEB.

### 3.7.1 Artificial OD Trails

Our first example uses a random graph with 100K OD trails, and  $5 \times 5$  grid and hierarchical reference graphs to simulate conventional road networks. Raw OD trails and corresponding road networks are shown in the leftmost figures of Figure 3.7 and Figure 3.6. The OD trails are first mapped to underlying road networks using shortest path algorithm, and then both road networks are divided into three hierarchies that are colored in red, blue and green, respectively. In the last,

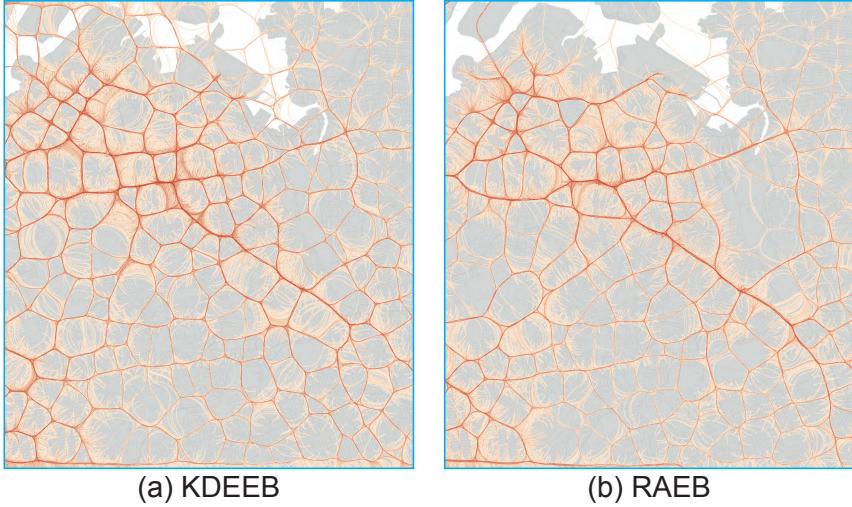


Figure 3.9. Fine scale density maps of NYC taxi trips in Queens zone generated by (a) KDEEB, and (b) RAEB.

we apply RAEB on the OD trails, with all graph drawing sizes set to  $1280 \times 1280$  and initial kernel sizes  $p_r$  set to 60.

In advance of KDEEB, RAEB allows to preserve route awareness by controlling parameter  $p_{ra}$ . To evaluate the effects of  $p_{ra}$ , we set  $p_{ra}$  to 0 - 3, and apply RAEB to OD trails on the grid road network as shown in the leftmost figure of Figure 3.7. The results are shown in Figure 3.7 (a) - (d). In (a),  $p_{ra}$  is set to 0, i.e., no route awareness will be preserved. In this case, RAEB should be equivalent to KDEEB with other parameters the same. The proof is demonstrated in (a), which shows smooth bundles similar to those in the original KDEEB [56]. To preserve level 1 routes, we set  $p_{ra}$  to 1, and the result is shown in (b). As expected, OD trails around the red lines are bundled together with bundles following red lines, while other OD trails are bundled with bundles spreading arbitrarily in the graph drawing. More route awareness can be preserved as we increase  $p_{ra}$  to 2 and 3, as shown in Figure 3.7 (c) & (d).

More importantly, route awareness  $p_{ra}$  can be used to preserve topology of underlying road network. Figure 3.6 (right) shows bundling result of RAEB applied to OD trails on a hierarchical road network that is shown in Figure 3.6(left). Here,  $p_{ra}$  is set to 2, such that to preserve both the red and blue lines. It is clear that the resulting bundles follow the lines, making it easy to trace OD trails. In contrast, KDEEB will generate the same result with that in Figure 3.7(a), as the underlying road network information is not used.

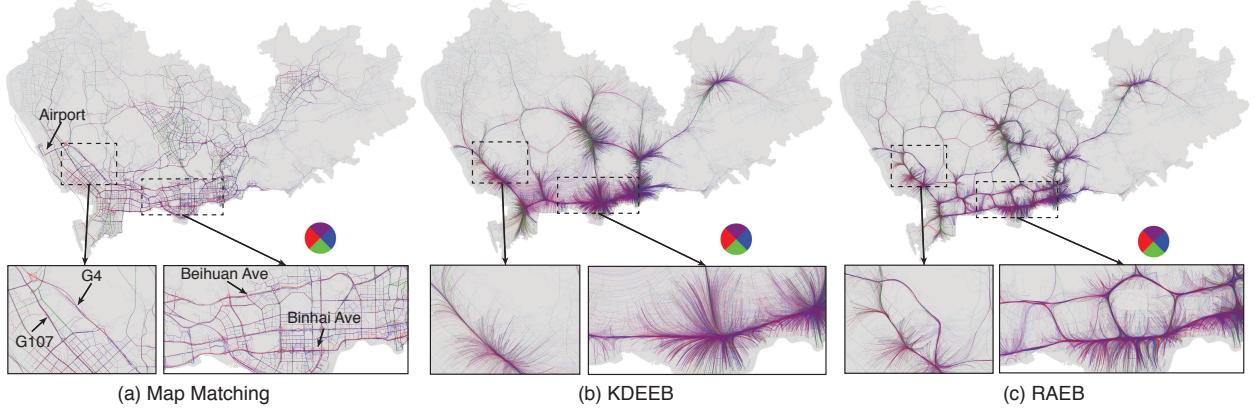


Figure 3.10. Density maps of Shenzhen taxi trips: (a) raw GPS records are mapped onto road network, (b) KDEEB bundles trips on close arterial roads together, while (c) RAEB preserves these roads. All lines are colored according to the OD directions.

### 3.7.2 New York Taxi Trips

In reality, a road network is usually much more complex, and OD trails are more dynamic than the artificial ones. In this experiment, we further compare RAEB with KDEEB using results generated from New York taxi trips.

The road network employed in this study is extracted from OpenStreetMap (OSM) within the boundary of Manhattan, Brooklyn, Queens, and Bronx zones, where most origins and destinations are located. The raw network consists of 133,154 nodes, 166,122 edges, and it is simplified to 97,336 routes. The OD trails studied in this experiment are 100K taxi trips extracted from one-month trip records. The original records consist of various attributes for each trip, including vendor id, pickup & dropoff times and positions, and fare amount, etc. We use the pickup & dropoff positions, and map it onto the road network using shortest path algorithm. Figure 3.8(a) presents a density map of the mapped taxi trips.

Figure 3.8(b - d) show bundling results generated by KDEEB and RAEB, with graph drawing sizes set to  $1080 \times 1440$ . The experiment is started with KDEEB recommended settings of kernel size  $p_r = 60$  (about 5% of graph drawing size) and  $p_n = 10$ . Figure 3.8(b) shows the bundling result, which clearly presents several main bundles. The bundles reveal primary traffic movements over the whole city. Yet in other ways, the results are heavily bundled, missing details of road network topology. To show more details, we measure initial kernel size considering both graph drawing and geometric property of road network as described in Section 3.5.1.  $p_r$  is reduced to 21. Figure 3.8(c)

shows the corresponding KDEEB bundling results. More bundles are formed, in line with arterial roads in New York. However, there are many bundles lying by non-existing roads, as those shown in the highlighted region in-between Manhattan and Brooklyn. This illustration can cause wrong impression of bridges connecting the zones. Figure 3.8(d) shows results generated by RAEB, with  $p_r = 21$  and route awareness  $p_{ra} = 1$ . The undesirable bundles are removed by our method.

Visualization of urban traffic should support multi-scale exploration – an important characteristic for movement data visualization. In the context of this work, a good edge bundling algorithm should provide smooth transitions of bundling results at different scales. We further evaluate multi-scale exploration by zooming into the blue region in Queens zone as in Figure 3.8(c) & (d), and applying KDEEB and RAEB on the OD trails. The fine scale bundling results are presented in Figure 3.9(a) & (b), respectively. More bundles are generated by both methods, as the same effect when zooming into maps. However, it is difficult to visually match bundles in Figure 3.8(c) with those in Figure 3.9(a). In comparison, bundles in Figure 3.8(d) can be more easily identified in Figure 3.9(d).

### 3.7.3 Shenzhen Taxi Trips

We further evaluate the performance of RAEB using taxi trips in Shenzhen, with road network extracted from OSM. In total, there are 161K nodes and 177K edges, and the edges are simplified to 51K routes. The OD trails are one-week taxi trip records. Unlike New York taxi records with only origin and destination positions, Shenzhen’s data contain a sequence of GPS positions recorded in every 20 seconds. We clean and extract in total 50K taxi trips from the records. The trips are further mapped onto the road network using ST-matching algorithm [81].

Figure 3.10(a) presents a graph of the map matching results in the size of  $2160 \times 1280$ . The lines of a trip are colored according to direction from the trip’s origin to destination. The graph shows that most taxi trips are recorded in the southern region of Shenzhen, which borders on Hong Kong and is more developed than other regions. The unevenly distributed taxi trips make it improper to use kernel size  $p_r$  recommended by KDEEB, which is over 100 as 5% of the graph drawing size. Instead, our method suggests  $p_r = 26$ , considering the road network topology as well.

Using this setting of  $p_r$ , both KDEEB and RAEB reaches stable states at iteration 8. Figure 3.10(b) & (c) show the corresponding results, respectively. Both methods create smooth bun-

dles in line with arterial roads in Shenzhen, indicating  $p_r$  measured by our method is preferable. Besides, nearly the same bundles are generated in regions other than south part, except that RAEB's result shows more details. More dissimilar results are shown in the highlighted areas. Expressways G4 and G107 lead traffic to Shenzhen airport in the left region, while Beihuan and Binhai avenues holds main traffic in the right one. KDEEB merges OD trails passing these roads, while our method splits the bundles. In short, RAEB generates results more similar to real-world scenarios.

## 3.8 Discussion

The experiments demonstrate that RAEB outperforms KDEEB in visualizing OD trails in urban traffic data. The first experiment with artificial data shows that the introduction of route awareness  $p_{ra}$  in RAEB distinguishes grid and hierarchical network structures. The second experiment presents the comparisons between KDEEB and RAEB using New York taxi data, which records only origin and destination positions. The results with different kernel size  $p_r$  and in multi-scales indicate that, (1)  $p_r$  should be measured based on not only graph drawing size, but also underlying road network topology; and (2) RAEB supports better multi-scale exploration. The robustness of RAEB is further demonstrated through the third experiment with Shenzhen taxi data, which records a sequence of GPS positions in every 20 seconds.

The advancements are achieved by parameter settings adopted in RAEB. This section presents detailed discussion on the parameters, followed by performance comparison between RAEB and KDEEB. In the end, we discuss about generality and limitation issues.

Parameters	KDEEB	RAEB
Route awareness ( $p_{ra}$ )	—	0 - Number of road hierarchies
Kernel size ( $p_r$ )	Graph drawing	Graph drawing and road network geometry
Bundling iterations ( $p_n$ )	10 - 15	—
Bundling stability ( $p_s$ )	—	$NMI$ between images in consecutive iterations
Bundling deviation ( $p_d$ )	—	of bundles to OD trails

Table 3.2. Main parameter adaptions in RAEB, in comparison with these in KDEEB.

### 3.8.1 Parameters

RAEB adapts and introduces new parameters to better visualize OD trails in urban traffic data. Table 3.2 presents an overview of these parameter settings in comparison with those in KDEEB.

- **Route awareness ( $p_{ra}$ ):** An input graph for KDEEB is straight lines connecting end nodes directly. As discussed in the first experiment with artificial OD trails, such an input graph cannot distinguish grid and hierarchical road networks. Instead, we introduce route awareness parameter  $p_{ra}$  to control layout of input graph. The first experiment also shows that  $p_{ra}$  can be manipulated to generate bundles representing levels of details of road network. If  $p_{ra}$  is set to zero, input graph will be straight lines and the results are the same with KDEEB; if  $p_{ra}$  is set to the highest number of road hierarchies, input graph will be OD trails in line with road network, and more roads can be preserved.
- **Kernel size ( $p_r$ ):** In the second experiment with New York taxi data, we show that the parameter kernel size  $p_r$  can control bundling coarseness. KDEEB computes  $p_r$  as 5% of graph drawing size, which will generate coarse bundles that can only reveal main traffic connections (Figure 3.8(b)). Instead, RAEB suggests  $p_r$  should be measured based on both graph drawing size and road network geometry. In this way, a smaller  $p_r$  is generated, and fine bundles that reveal details of street networks are generated (Figure 3.8(c) & (d)). The advantage of our approach is further demonstrated using Shenzhen taxi data, which exhibits imbalanced distribution of taxi trips in the city.
- **Bundling iteration ( $p_n$ ):** KDEEB presets bundling iteration  $p_n$  to a fixed number in between 10 - 15, such that to terminate the bundling process. The choice of a suitable  $p_n$ , however, needs much experience and varies among different people. Adjustments in other parameters may also require fitting of the parameter. RAEB discards this subjective parameter.
- **Bundling stability  $p_s$ :** Instead, RAEB introduces a new parameter – bundling stability  $p_s$ , to control bundling process termination.  $p_s$  is measured as normalized mutual information (*NMI*) between images generated in two consecutive bundling iterations. *NMI* is a common metric in image processing field, thus can be easily integrated. When  $p_s$  reaches a predefined threshold, which means that visually a stable structure is generated, the bundling process will be automatically terminated.
- **Bundling deviation  $p_d$ :** While  $p_s$  measures bundling stability of a bundling method, it still lacks a method to quantitatively compare multiple ones. The parameter bundling deviation  $p_d$  is introduced to fill the gap.  $p_d$  is measured as average Frechet distance of bundles to

Graph	Edge Samples	$p_n$	Time (sec.)		Deviation (pixel)	
			KDEEB	RAEB	KDEEB	RAEB
Random (grid)	4.6M	13	40.3	50.7	18.37	12.58
NewYork	3.1M	11	34.3	42.9	15.40	9.88
Shenzhen	1.3M	8	13.8	22.8	13.71	10.53

Table 3.3. Statistics comparison of KDEEB and RAEB for datasets used in the experiments.

corresponding OD trails. The measurement is widely employed in geographical science, and we compare KDEEB and RAEB regarding the parameter below.

Some of these parameters can be automatically computed, including  $p_r$  and  $p_d$ , while settings of  $p_{ra}$  and  $p_s$  can be consistent across different input data. From the experiments, we recommend 1 for  $p_{ra}$  (corresponding to top 5% ranked routes), and 75% for  $p_s$ .

### 3.8.2 Performance

Table 3.3 shows comparison of KDEEB and RAEB for the datasets used in the experiments. Both methods are implemented in Java running on a MacBook Pro with a 3.1 GHz Core i7 and Radeon Pro 560 graphics. We employ the same kernel size  $p_r$  and bundling iterations  $p_n$  measured by our approach for both methods.

KDEEB’s bundling process include sampling, splatting, advecting and smoothing steps. Bundling time for each iteration is mainly determined by number of edge samples and kernel size. Since kernel size measured by our approach is generally smaller than that by KDEEB, bundling time is reduced. The total running times for KDEEB methods are comparable with those in the original work [56]. RAEB requires an additional step of bundling stability measurement, which is determined by the graph drawing size. This adds in a running time of less than 1 seconds in each iteration for all three experiments. The measurement is computed in every bundling iteration, thus we can shorten time by reducing the measurement frequency. All the steps can be further accelerated using GPU e.g. CUDA [113].

On the other side, bundles generated by our method are more close to OD trials, as revealed by comparison regarding bundling deviation. From the table, RAEB reduces about 1/3 deviations of those in KDEEB for artificial and New York taxi datasets. The improvement is smaller for

Shenzhen taxi data, though graph drawing in Shenzhen experiment is bigger. This is probably because taxi trips are concentrated in a small area in the south of Shenzhen. Nevertheless, the result image by RAEB shows more improvements than that by KDEEB (Figure 3.10(c) vs (b)).

### 3.8.3 Applicability and Limitations

The experiments have shown the applicability of RAEB for bundling and visualizing OD trails in urban traffic using real-world taxi data. RAEB introduces new parameters of route awareness  $p_{ra}$  and bundling deviation  $p_d$ , and adapts existing measurement of kernel size  $p_r$ , to better fit road network topology. The approach can be extended to other types of urban traffic data, such as commuter trips using public transportation, or people traces derived from call detail records. These dataset consists of both OD trails and underlying road network. Through advanced map matching algorithms, urban traffic's real paths on a road network can be recovered. However, the parameters are not applicable to general graphs with information of only node connections, except for bundling stability  $p_{bs}$ .

Edge bundling methods face a common issue of unrealistic bundle lines. Specifically for OD trails in urban traffic, bundle lines should not be far away from real paths. RAEB mitigates the issue with route awareness  $p_{ra}$  for controlling edge layout of input graph, and bundling stability  $p_{bs}$  for generating stable results. The first experiment shows that  $p_{ra}$  can preserve certain routes on demand, while the second experiment reveals that RAEB supports better multi-scale exploration. Experimental results of bundling deviations  $p_d$  further proves the advantage of RAEB over KDEEB. Nevertheless, these parameters can only constrain bundle layout in certain extent, as resulting bundle lines are still displaced from OD trials. In scenarios requiring precise positions, such as query movements passing through a waypoint [62] or road segment [119], visual hints are required to remind the displacement.

Bundles generated by RAEB can show clear traffic patterns, meanwhile preserve road network topology better than KDEEB in a city. These main goals of this work have been successfully achieved, as demonstrated by the experiments. On the other hand, RAEB is not able to reveal detailed traffic patterns such as directional movements. Direction is of recognized importance for studying OD trails in urban traffic [130]. This limitation can be addressed by including directional information in density estimation or edge advecting functions, as described in CUBu [113]. Besides, RAEB is an image-based edge bundling method, constraining its applicability for 2D ren-

dering only. Applications such as to bundle movements [66, 110] and fiber tracts [55] in 3D, and graphs in virtual reality [65], need spring- or geometry-based bundling methods.

### 3.9 Conclusion and future work

OD visualization is challenging – no universally good technique is available for visualizing arbitrary ODs [6]. In particular, this work studies OD trails in urban traffic. We identify inconsiderate settings of existing kernel density estimation edge bundling (KDEEB) method when applied in the scenario. This is because KDEEB was developed for general graphs such as airlines and migrations, while ignores data nature of urban traffic data. These inconsiderate settings can cause domain-specific problems such as no preservation of road network topology and poor support of multi-scale exploration. We contribute to the field with a new technique, namely route-aware edge bundling (RAEB). The bundling process is complemented with preprocessing that generate more proper input edge layout, and evaluation that create more stable visual results. A series of parameter adaption and introduction are made to better cater to geometric properties of OD trails and road networks. We compare RAEB with KDEEB in regarding to bundling time and deviation using artificial and real-world taxi data. The experiment results show that RAEB outperforms KDEEB in reducing bundling deviation meanwhile achieving comparable computation speed.

Even though RAEB is specifically designed for bundling OD trails in urban traffic, some parameters such as bundling stability  $p_{bs}$  can be applicable to other iterative edge bundling methods, including SBEB [30] and FDEB [52]. We would like to try integrating these parameters in the bundling methods, such that to make them more controllable. We also realize many computation tasks in RAEB can be parallelized, thus bundling efficiency can be improved by GPU acceleration. It would be worthy to reimplement RAEB in GPU, similar to CUBu [113]. Besides bundling control and efficiency issues, how to access bundling quality and faithfulness are the main challenges hindering the applicability of edge bundling methods [69]. Our experiences in developing RAEB suggest that in specific application scenarios, these challenges can be solved with domain knowledges. On the basis of treating OD trails mapped onto road network as ground truth, our proposed parameter of bundling deviation  $p_d$  is a quality and faithfulness metric in certain extent. The parameters is derived from Frechet distance that is a common metric in geographical science. In the future, we would like to apply RAEB in other fields such as to visualize neural connections

in brain [13, 129]. Adaptons are most likely required, and we envision close collaborations with domain experts can facilitate the development.

## CHAPTER 4

# VISUAL INTERPRETATION OF RECURRENT NEURAL NETWORK ON MULTI-DIMENSIONAL TIME-SERIES FORECAST

Recent attempts at utilizing visual analytics to interpret Recurrent Neural Networks (RNNs) mainly focus on natural language processing (NLP) tasks that take symbolic sequences as input. However, many real-world problems like environment pollution forecasting apply RNNs on sequences of multi-dimensional data where each dimension represents an individual feature with semantic meaning such as  $PM_{2.5}$  and  $SO_2$ . RNN interpretation on multi-dimensional sequences is challenging as users need to analyze what features are important at different time steps to better understand model behavior and gain trust in prediction. This requires effective and scalable visualization methods to reveal the complex many-to-many relations between hidden units and features. In this work, we propose a visual analytics system to interpret RNNs on multi-dimensional time-series forecasts. Specifically, to provide an overview to reveal the model mechanism, we propose a technique to estimate the hidden unit response by measuring how different feature selections affect the hidden unit output distribution. We then cluster the hidden units and features based on the response embedding vectors. Finally, we propose a visual analytics system which allows users to visually explore the model behavior from the global and individual levels. We demonstrate the effectiveness of our approach with case studies using air pollutant forecast applications.

### 4.1 Introduction

Recurrent neural networks (RNNs) have been widely applied in various natural language processing (NLP) tasks such as machine translation and sentiment analysis. Benefiting from the capacity to model sequential data, RNNs have been extended to other domains of sequential data beyond NLP, such as weather forecast [15, 102, 127] and air pollutant prediction [72, 89, 139]. Compared to NLP tasks that take latent embeddings as inputs, the input features in these applications are usually multi-dimensional time series where each dimension has its own semantic meaning. For example, in air

pollutant forecast, RNN models are widely adopted by domain experts where input sequences are hourly recorded series of high-dimensional pollutants (*e.g.*,  $SO_2$ ) and meteorology features (*e.g.*, *wind speed*).

Despite the competitive performance of RNNs, the lack of understanding of their internal mechanisms makes the models untrustworthy and further limits their extension to other domain applications. During model development, the domain experts usually want to better understand the models' forecast. They tend to learn whether the model's behavior confirms any hypotheses according to existing domain knowledge, which helps gain confidence in prediction for decision making. On the other hand, the experts also aim to identify the patterns that have not been observed before by exploring different cases to enrich their knowledge of the domain problem. In addition, understanding RNNs also helps model designers to choose appropriate model architecture and hyper-parameters.

Existing work on RNN interpretation such as LSTMVis [106] and RNNVis [85] mainly focus on NLP tasks. Both of these two works interpret hidden units by providing their relevant words as language contexts. However, existing techniques cannot be directly applied in RNNs for high-dimensional time-series forecasts. First, the high dimensionality of the input sequence makes it difficult to discover relationships between features and hidden states. Since different features at various timestamps are correlated, traditional techniques are not applicable as they are not designed to reveal how feature importance changes over time, which hinders the domain experts' ability to analyze if the prediction is consistently generated. Analyzing this complex temporal multi-dimensional data requires an effective visualization design that can reveal both cross-dimension data relationships and the model's temporal behavior. Moreover, in real-world applications, the input dimension size may be from hundreds to thousands, and the hidden unit size of the RNN models can also be large. This requires the visualization methods to have good scalability in order to demonstrate the distribution and complex relationships of hidden units and input features.

To address these challenges, we propose MultiRNNExplorer, a visual analytics system to help domain experts understand RNNs in high-dimensional time-series forecasts from two aspects: model mechanism and feature importance. To understanding model mechanism, we estimate the overall response from hidden units to features and generate response embedding for both hidden units and features. We then cluster the hidden units and features respectively to reveal the high-level knowledge captured by the models. To measure the feature importance, we calculate the gradient for all features at all timestamps with respect to a given case. Then a visualization interface with

coordinated views is designed, enabling the users to interactively explore the model behavior.

Our contributions can be summarized as follows:

- A new method for estimating neuron response to multi-dimensional time-series data by measuring hidden unit response distribution on different value ranges of input features.
- A visual analytics system that help users to explore, interpret, and compare RNNs on multi-dimensional time-series data.
- Several case studies on air pollutant datasets to demonstrate the effectiveness of the proposed system and the insights revealed during exploration.

## 4.2 Related Work

### 4.2.1 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) is a class of neural networks that contains feedback connections [49]. Compared with fully connected neural networks, this architecture is capable of processing temporal data and learning sequences. Many RNN variants have also been developed in the past decades. One typical work is Long-Short Term Memory (LSTM) [50]. By integrating several gate functions and appending an additional cell state to the vanilla RNN, LSTM avoids the gradient vanishing problem when processing the long sequences. A similar but computationally more efficient approach is the Gated Recurrent Unit (GRU) [20], which simplifies the model architecture by using only two gate functions: update gate and reset gate. To better utilize more data along the sequence, the bi-directional RNN [99] is proposed to capture both past and future information to improve prediction. Greff et al. proposed a comprehensive survey [42] to summarize and compare these RNN variants in order to guide model selection under different scenarios.

RNN-based models have not only been proven successful in performing Natural Language Processing (NLP) tasks such as machine translation [109] and question answering [48], but have also been adopted in broader domains such as weather forecasting [127] and environmental factor prediction [19]. For example, Oprea et al. proposed an RNN-based architecture to forecast  $PM_{2.5}$  air pollutants [89]. Compared with the NLP domain, these critical areas not only require good model performance but also need humans to understand how a prediction is generated and whether the results are trustable for guiding further decision makings [75]. However, understanding RNNs in

such scenarios is challenging. On the one hand, as each input dimension is usually informative and usually indicates an interpretable factor compared with word embeddings used in NLP, analyzing which features have the most impact on prediction results is important. On the other hand, users also need to explore and understand how the hidden states evolve over time in order to reveal the underlying working mechanism of the RNN along the sequence. We propose MultiRNNExplorer to better understand and interpret RNN models, especially when the model input is multi-dimensional as with the above critical areas.

### 4.2.2 Machine Learning Interpretation

In recent years, many machine learning interpretation methods have been developed, which can mainly be categorized into two groups: model reduction and feature contribution.

Model reduction methods usually learn a surrogate model to approximate the original complex model. The surrogate model is usually simple and interpretable, such as linear regression [96] and decision trees [23]. Depending on the ways of approximating the original model’s behaviors, there are three main ways to conduct model reduction: decompositional, pedagogical, and eclectic [2]. Decompositional methods are usually model dependent and simplify the original model structure, such as the layer and weights of the neural network. Pedagogical methods only utilize the input and output information to mimic the original model. Eclectic methods are either a combination of the previous two approaches or are distinctively different from them. Though model-reduction-based methods are flexible and easy to understand, it is questionable whether or when the surrogate model truly reflects the original model’s behaviors. We thus discard this approach in our work.

Feature contribution methods help users understand the relationships between input features and the output prediction. They usually assign each feature an importance score to indicate how it impacts the final prediction. One classical work is Partial Dependence Plot (PDP) [35], which depicts how feature value changes affect predictions. A PDP is usually visualized as a line chart in which the x-axis represents feature values and the y-axis represents prediction possibilities (partial dependence scores). For each feature, its partial dependence scores are usually calculated by iteratively fixing the feature input of all the data points to a certain value and getting the average prediction. One recent work, SHAP [82], also calculates feature attribution, but from a local perspective. SHAP is based on the ideas of Shapley Values from game theory, which calculates the feature contribution of a data instance by comparing it with a set of reference data points. Compared

with global solutions, it is more consistent and locally accurate. One major limitation of feature contribution methods is that they are computationally expensive. In this work, we alleviate this problem by adopting a pre-processing stage before interactive exploration.

Apart from the above work that focuses on describing the input-output relationship, some work focuses on understanding and analyzing the internal working mechanism of RNNs, especially the hidden layer behaviors, by utilizing visualization techniques. Karpathy et al. first conducted some explorative studies on hidden cell activation on different sequence items [60]. LSTMVis [106] further analyzed and compared the activation changes of each individual cell to demonstrate that different cells may capture different language patterns over time. Ming et al. developed RNNVis [85] to depict the co-clustering patterns between hidden states and input words. There is also some other work that focuses on a particular model or domain. For example, Seq2Seq-Vis [105] mainly focuses on the sequence-to-sequence model and RetainVis [64] enables experts to analyze electronic medical data using a RNN model. Though these methods show how the relationships between hidden units and input data change over time, they fail to capture the importance evolution of each individual input dimension, which prevents them being adopted in critical areas such as finance and environmental factor forecasting. We aim to solve this problem by revealing the activation sensitivity of hidden units to different features and highlighting what features mostly affect the final prediction the most over time.

## 4.3 Application and Models

### 4.3.1 Application

In this chapter, we focus on a particular type of regression task on multi-dimensional time-series data: air pollutant forecast of **target pollutants** at **target locations**. We collaborate with a group of domain researchers who use RNNs to conduct air pollutant forecasting. Specifically, they choose 16 air quality monitoring stations in Hong Kong with the stations' locations as the **target locations** and select  $PM_{2.5}$ ,  $PM_{10}$ ,  $NO_2$ ,  $SO_2$ , and  $O_3$  as the **target pollutants**. Though the RNNs can provide high accuracy, the researchers are more interested in understanding why the models make certain predictions so that they can decide whether to trust the models for decision making based on their domain knowledge. We thus develop MultiRNNExplorer to help the domain researchers explain the RNNs' behavior on their temporal multi-dimensional air pollutant dataset.

### 4.3.2 Data Description

The dataset includes hourly observations of the air pollutant and meteorology data from different air quality monitoring stations in China from Jan. 01 2015 to Dec. 31 2018. The features are listed in Table 4.1.

Category	Feature type
Air pollutant	$PM_{2.5}, PM_{10}, NO_2, NO_x, SO_2, CO, O_3$
Meteorology	<i>Wind Speed(Wind), Wind Direction(WD), Dew Point(DP), Relative Humidity(RH), Temperature(Temp), Sea Level Pressure(SLP), Station Pressure(SP), Cloud Cover(CC)</i>

Table 4.1. There are two types of features taken as input: air pollution and meteorology.

Similar to other work on air pollutant forecasts [135], when forecasting the air pollutants at a target location, we divide the regions around the target location into different spatial partitions and aggregate the data observed by all the stations within each group in order to generate features. Specifically, as shown in Fig. 4.5B, we divide the nearby regions into five non-overlapping rings centered at the targeted location with radii of 10km, 30km, 100km, 200km, and 300km. Each ring is further divided into 8 sectors with the same angle such that the whole area around the target location is partitioned into 40 sub-regions. For each sub-region, we use the air pollutant and meteorology data observed by the monitoring stations located in the corresponding sub-region as the features. If there are multiple monitoring stations in a sub-region, we aggregate their data and use mean values as the features. In this way, each feature can be identified as a triplet (*distance, direction, feature\_type*); for example, (10km, E,  $NO_2$ ) indicates  $NO_2$  concentration observed at 10km away *East* of the target location. For each time step, we use a vector to represent all the features where each dimension indicates a feature triplet. In this way, the data within a time period can be represented as a multi-dimensional sequence.

### 4.3.3 Models Description

RNNs take a sequence as input with fixed length  $T$ :  $\{x_0, x_1, \dots, x_{T-1}\}$  and predict the value at timestamps equal to or greater than  $T$ , where  $x_t \in \mathbb{R}^m$ . A hidden state  $h_t$  is updated according to the input of timestamp  $t$  and previous hidden state  $h_{t-1}$ . In vanilla RNNs, the hidden states are updated by:

$$h_t = \sigma(Wh_{t-1} + Vx_t) \quad (4.1)$$

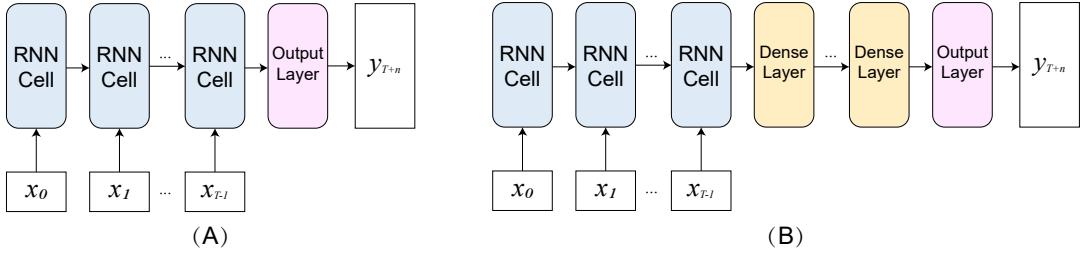


Figure 4.1. RNN Architectures considered in our experiments: A) RNN: the RNN layer is directly connected to the output layer; B) RNN-Dense: adding dense layers between the RNN layer and the output layer.

Where  $W$  and  $V$  are weight matrices and  $\sigma$  is the *tanh* function which constrains the output of the hidden states to  $(-1, 1)$ . We consider three types of architectures shown as Fig. 4.1. **RNN**: the final timestamp hidden state is directly connected to the output layer (Fig. 4.1A). **RNN-Dense**: there are several dense layers between the final timestamp and output layer (Fig. 4.1B). In addition to vanilla RNNs, we also consider two variants: GRU and LSTM, which mitigate the gradient vanishing issue and enable the models to memorize long-term information by adding the “gating” mechanism.

## 4.4 System design

In this section, the requirement and tasks are discussed. Over the past 12 months, we closely worked with two domain researchers in urban air quality analysis and forecasting. One researcher (R1) studies the atmospheric diffusion of air pollutants and has interest in what machine learning model learns. The other researcher (R2) is working on air pollutant forecasts through machine learning techniques. Both of them have strong domain experience in air quality forecasting.

### 4.4.1 Task analysis

During the discussions, three general goals were distilled: G1: Understand the RNN model behavior/mechanism in high-dimensional forecasts. G2: Understand the feature importance. G3: Support case-based exploration.

To fulfill the these analytics goals, we have specified the following analytical tasks:

**T1: Encode hidden state statistics.** Hidden states, a direct reflection of a model’s intermediate

results, are critical for revealing the information captured by a model (**G1, G2**). Visualizing hidden state statistics can provide a holistic picture of a model’s capacity and behavior.

**T2: Measure and encode feature importance at multiple scales.** The visual analytics system should allow users to explore feature importance at different scales(**G2**). For example, the overview level ranks and presents feature importance summarized from the whole dataset. The individual level will focus on importance of single case.

**T3: Analyze the response between features and hidden states.** Measuring the response relationship between features and hidden states is the key factor in revealing what patterns are captured by the model (**G1**). Targeting at the complicated many-to-many relationship, the hidden states as well as the features should be clustered to alleviate the burden on end users.

**T4: Support temporal analysis.** One major advantage of RNNs is that they can capture time-dependent sequence information. Showing what information is preserved along the sequence and what information is discarded helps users better understand how the temporal information is utilized by the model(**G2, G3**). In addition, users can identify the most critical time step that causes a significant change in the model’s prediction.

**T5: Identify data clusters and outliers.** To support case-based reasoning, users need to first obtain a data overview by identifying the data clusters and outliers (**G1, G3**). This provides concrete examples to guide users in further exploring the data of interests. Users can also inspect the outliers that have distinct prediction results to detect if the model behaves incorrectly according to certain domain knowledge.

**T6: Explore the case-based model behaviour.** The system needs to enable the users to explore how a model behaves for individual cases such as build the correlation between feature trend and feature importance trend(**G3**). Since hundreds of temporal features are taken as input for each case, the effective summary is required.

#### 4.4.2 System Overview

We implement MultiRNNExplorer as a web-based system using Flask, VueJS, and D3. The system consists of three modules: 1) preprocessing, 2) analysis, and 3) visualization. With chosen models, the preprocessing module generates the raw data that need to be analyzed, including estimating the response relationship and feature importance. We then apply various data analysis techniques such

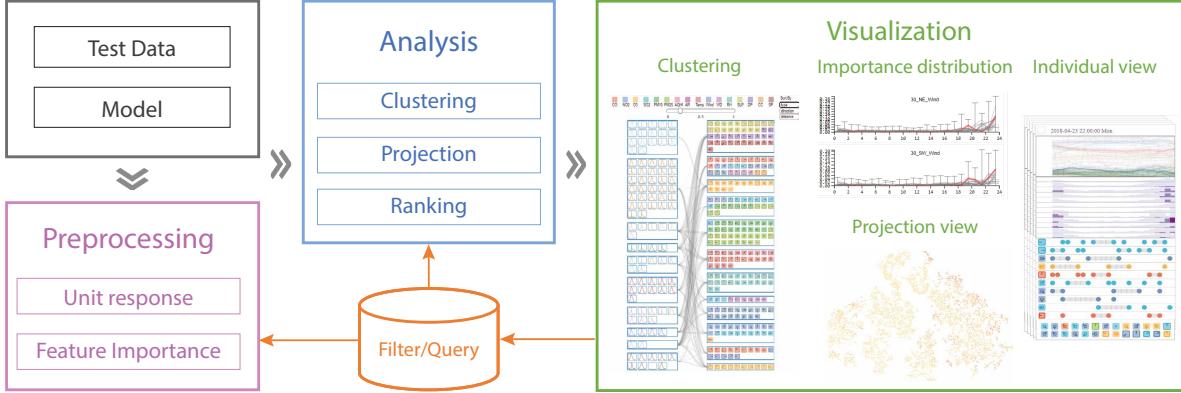


Figure 4.2. The system overview. There are three major modules in our system: Preprocessing module, Analysis module and Visualization module.

as clustering, projecting, and ranking in the analysis module to provide the data structure required by the visualization module. The visualization module integrates coordinated views to support interactive interpretation of and reasoning about the model behavior at different perspectives.

## 4.5 Model Interpretation

This section first describes how we analyze the relationships between features and hidden states (**T3**). Specifically, we propose an efficient method to calculate how sensitive each hidden state is to certain feature changes and apply a clustering method to group response relationship patterns for better scalability. This provides users an overview on how the model categorizes different features and perturbing features to what value ranges may largely affect model behaviors. We also introduce a gradient-based method to identify the most important features that can impact the prediction over time (**T2, T4**). This provides another perspective on analyzing how feature importance changes along the sequence. These two approaches are complementary to each other in enabling users' understanding and explanation of model behaviors.

### 4.5.1 Relationships between Hidden States and Features

To measure how feature changes can affect hidden states (**T4**), one common approach is perturbing feature values and measuring how the hidden state distribution changes compared with the original data. However, perturbation-based methods are usually time-consuming and not applicable when different features are correlated. Inspired by [108], we adopt another method that directly

compares the hidden state distributions of different feature value ranges. This approach is computationally efficient and provides a good approximation for whether a hidden state is sensitive to feature changes. This section introduces how we generate the hidden state distribution for different value ranges of each feature and how we quantitatively measure the relationship strength between features and hidden states based on the generated distribution.

## Hidden State Distribution

As discussed in Sec. 4.3.2, the model input is a sequence of features  $X = \{x_0, x_1, \dots, x_{T-1}\}$  where  $x_t$  indicates a multi-dimensional feature vector at time step  $t$ . Each feature dimension is denoted as  $x_t^f$  in which  $f$  represents a feature triplet (*distance*, *direction*, *feature\_type*) at time step  $t$ . Similarly, we use  $H = \{h_0, h_1, \dots, h_{T-1}\}$  to indicate the hidden state sequences at different time steps where  $h_t = \{h_t^0, h_t^1, \dots, h_t^{D-1}\}$  indicates the hidden state distribution at time step  $t$  and  $D$  denotes the hidden unit size. As  $h_t$  is computed by feeding  $x_t$  into an RNN model  $L$ , we denote  $h_t = L(x_t)$ . Considering a dataset  $\mathbb{X} = \{X_0, X_1, \dots, X_{N-1}\}$  consisting of  $N$  sequences, we can collect a feature vector set  $\mathbb{V} = \{x \mid x \in X, X \in \mathbb{X}\}$  where  $|V| = N \times T$ . Based on the value ranges of a feature  $f$ , we can further divide  $\mathbb{V}$  into different groups  $\mathbb{V}_g^f = \{x \mid \Theta_g^{lower} \leq x^f < \Theta_g^{upper}, x \in \mathbb{V}\}$  where  $\Theta_g^{lower}$  and  $\Theta_g^{upper}$  denote the feature range thresholds of a group  $g$ . In this chapter, we set the number of groups to be 3 where the thresholds are the 25<sup>th</sup> and 75<sup>th</sup> percentiles of each feature. We denote these three groups as  $\mathbb{V}_{perc<0.25}^f$ ,  $\mathbb{V}_{0.25 \leq perc < 0.75}^f$ , and  $\mathbb{V}_{perc \geq 0.75}^f$ . As we can obtain the hidden states by feeding the data into the RNN model, we can compute the corresponding hidden state set  $\mathbb{H}_g^f = \{L(x) \mid x \in \mathbb{V}_g^f\}$  for a feature group  $\mathbb{V}_g^f$ . In this way, the distribution of the  $j^{th}$  hidden unit for feature group  $\mathbb{V}_g^f$  can be denoted as  $\mathbb{H}_g^{j,f} = \{h^j \mid h \in \mathbb{H}_g^f\}$ . Measuring the distribution of  $\mathbb{H}_g^{j,f}$  enables us to compare the outputs of different hidden units when a feature value falls into a certain range and infer if these hidden units are sensitive to feature value changes. For example, Fig. 4.3 shows the distribution of the 92<sup>th</sup> and 93<sup>th</sup> hidden units for feature  $PM_{2.5}$  and  $SO_2$  respectively. We can see that the 92<sup>th</sup> hidden unit has distinct distributions for different value ranges on feature  $PM_{2.5}$ . Meanwhile, for feature  $SO_2$ , the distributions look identical. This indicates that the 92<sup>th</sup> hidden unit is more sensitive when the value of  $PM_{2.5}$  changes compared with  $SO_2$ . Similarly, we can observe that the 93<sup>th</sup> hidden unit is more sensitive to  $SO_2$  changes, which indicates that different hidden units can capture distinct feature patterns.

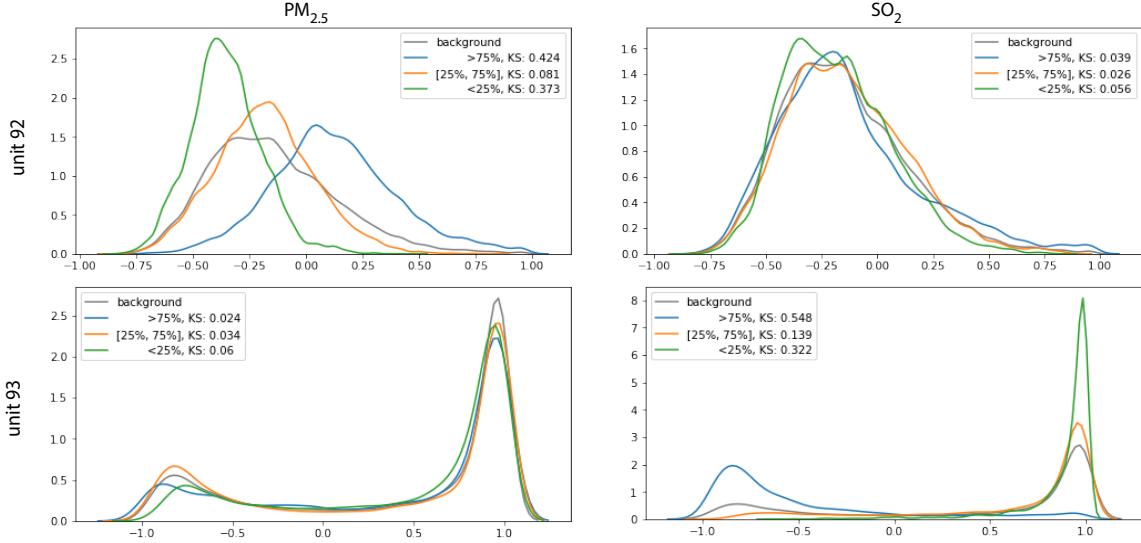


Figure 4.3. Compare the response of hidden units(92 and 93) to features ( $PM_{2.5}$  and  $SO_2$ ).

### Relationship Strength Estimation

We estimate the relationship strength of a hidden unit with a feature by measuring the distances between the hidden unit distributions of different feature value ranges. To measure distribution distance, we apply Two-sample Kolmogorov Smirnov (KS) statistics which can be presented in following formulation:

$$KS(S1, S2) = \max_{sup_x} (|F_{S1}(x) - F_{S2}(x)|) \quad (4.2)$$

where the  $sup_x$  is the supremum of the set of distances,  $F_{S1}$  and  $F_{S2}$  are the cumulative empirical distribution functions of the first and the second sample respectively, and  $sup$  is the supremum function. Given significance level  $\alpha$  (generally 0.05) the null hypothesis of two samples having different contributions, the reject co-efficient can be calculated as follows:

$$Rej(S1, S2) = c(\alpha) \sqrt{\frac{|S1| + |S2|}{|S1||S2|}}, c(\alpha) = \sqrt{-\frac{1}{2} \ln \alpha} \quad (4.3)$$

Based on the KS statistics, the distance between two samples can be measured as follows:

$$Dis(S1, S2) = \begin{cases} KS(S1, S2), & \text{if } KS(S1, S2) > Rej(S1, S2) \\ 0, & \text{otherwise} \end{cases} \quad (4.4)$$

To quantitatively measure the relationship strength between a hidden unit and a specific input feature, we compare the hidden unit distribution of data in different feature ranges with the distri-

bution of all the data. A larger difference indicates a stronger relationship as the hidden unit will generate different values when the feature value changes. As shown in Fig. 4.3, the ks-statistics of unit 92- $PM_{2.5}$  and unit 93- $SO_2$  are significantly larger than the other two combinations, indicating the statistics can effectively measure the distribution difference. Specifically, the relationship strength between the  $j^{th}$  hidden unit and feature  $f$  can be measured as the maximum ks-statistics among all different feature selections:

$$\begin{aligned} RS(j, f) = & \max(Dis(\mathbf{H}^{j,f}, \mathbf{H}_{perc<0.25}^{j,f}), \\ & Dis(\mathbf{H}^{j,f}, \mathbf{H}_{0.25 \leq perc < 0.75}^{j,f}), \\ & Dis(\mathbf{H}^{j,f}, \mathbf{H}_{perc \geq 0.75}^{j,f})) \end{aligned} \quad (4.5)$$

#### 4.5.2 Hidden Unit and Feature Clustering

Another major challenge for interpreting RNN models on multi-dimensional sequential data is scalability. RNN models usually contain hundreds to thousands of hidden units for each layer, which makes it ineffective to display the activation distribution of every hidden unit to users. To address this challenge, previous work on visual interpretation of machine learning models usually use clustering [77, 85] or sampling [93] techniques to reduce the number of visual elements displayed. In this work, we choose clustering methods over sampling since clustering can better preserve the hidden units' response relationship to features. It also provides a good summary of the knowledge that the model learned.

With the measurement of unit response, we can generate a 2D table with the size of  $D \times M$ , where  $D$  and  $M$  are the size of hidden units and features respectively. The cell of  $j^{th}$  row and  $k^{th}$  columns is the response of hidden unit  $h^j$  to feature  $f^k$ :  $RS(j, f^k)$ . Then we can define the response embedding vector for both features and hidden units. For any feature  $f^k$  and hidden unit  $j$ , the response embedding vectors are  $vec_{f^k} = [RS(0, f^k), RS(1, f^k), \dots, RS(D-1, f^k)]$  and  $vec_j = [RS(j, f^0), RS(j, f^1), \dots, RS(j, f^{M-1})]$ , which are the specific columns and rows respectively.

To analyze the relationship between hidden units and features, Yao et al. [85] used a bipartite graph to model the many-to-many relationship and used co-clustering algorithms [26] to group hidden units and input features simultaneously. We test co-cluster techniques: Spectral Co-clustering(SCoC) as well as other techniques including Agglomerative Clustering(AC) and Spectral Clustering(SC) on our dataset. The clustering methods other than SCoC take response embedding vectors as input

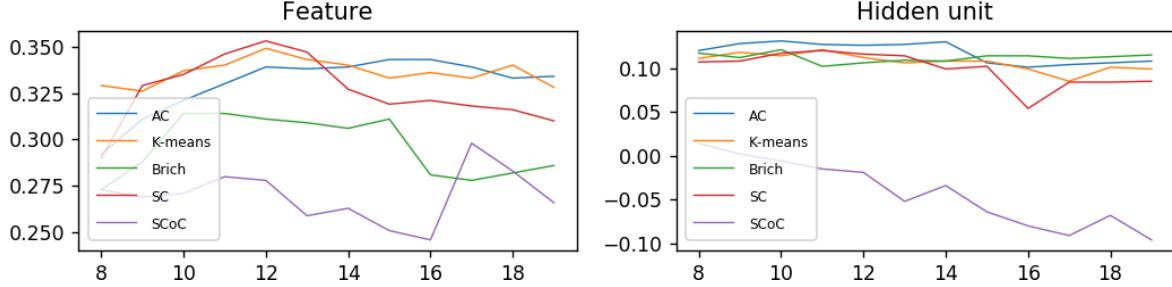


Figure 4.4. Cluster score with different cluster number. Left: feature cluster. Right: hidden unit cluster. The horizontal axis represents the cluster number, the vertical axis represents the cluster score.

to cluster features and hidden units respectively. To rank the performance of different clusters with different cluster numbers, we use the Silhouette Coefficient [97] to evaluate the quality of the clusters. Silhouette Coefficient ranges from -1 to +1, with higher values of this coefficient meaning the cluster quality is more appropriate.

Fig. 4.4 shows cluster quality for features (left) and hidden units (right). We found that the Spectral Co-clustering method has a low Silhouette Coefficient score because it keeps creating a one-to-one relationship between the feature cluster and the hidden units cluster. In this case, it can be found that Agglomerative Clustering with cluster number of 12 and K-Means with cluster number of 10 show the best performance for feature and hidden units respectively. With the Silhouette Coefficient, our system can automatically choose the clustering algorithms and cluster number. Users can also manually choose different clustering algorithms and change the number of clusters based on their analysis requirement.

The clustering results can be modeled as bipartite graph  $\mathbb{G} = (\mathbb{V}_H, \mathbb{V}_F, \mathbb{E})$ , where  $\mathbb{V}_H$  is the hidden unit cluster set and  $\mathbb{V}_F$  is the feature cluster set.  $\mathbb{E}$  indicates the weighted edge set between unit clusters and input dimension clusters with the weight of  $E_{H,F} = \sum_{h \in H} \sum_{f \in F} RS(h,f)$  where  $H \in \mathbb{V}_H$  and  $F \in \mathbb{V}_F$ .

This bipartite graph of features and hidden units can help users understand the information captured by different hidden unit clusters by examining which feature clusters have strong relationships with them.

### 4.5.3 Local Feature Importance

Inspired by back-propagation in machine learning, we conduct the individual level analysis based on the local gradient which is used to present the word saliency in NLP tasks [71]. Given the output of feature  $y^l$ , we use the local gradient with respect to feature  $x_t^k \in x$  to present the feature importance as:

$$w(y^l, x_t^k) = \left| \frac{\partial(y^l)}{\partial(x_t^k)} \right| \quad (4.6)$$

The absolute value of gradient  $w(y^l, x_t^k)$  indicates the sensitiveness of  $x_t^k$  to the final decision of  $y^l$  with the given input sequence of  $x$ . This measurement shows how much the specific feature at a specific time contributes to the final output [71]. However, for input  $x$  with the length of  $T$ , the total number of all feature importance scores is  $N \times T$  which causes difficulty in showing the overview. To address this challenge, we leverage the clustering result from Sec.4.5.2 and define the cluster importance of features as:

$$W(y^l, H_t^i) = \sum_{x_t^k \in H_t^i} |w(y^l, x_t^k)| \quad (4.7)$$

Thus, the size of the cluster importance for all timestamps is  $C \times T$  where  $C$  is the number of clusters ( $C < N$ ).

## 4.6 Visualization Design

In this section, we introduce the visual design based on the design tasks discussed in Sec.4.4.1. As shown in Fig. 4.5, the visual analytic system consists of six coordinated views. Starting from the configuration panel Fig. 4.5B, the users are able to select the target feature and the model to be analyzed. The region partition will be shown as Fig. 4.5B after the model is selected. To support exploring the model mechanism, the Cluster View is displayed to summarize the hidden units' response to the features (Fig. 4.5A) and the Feature Importance View (Fig. 4.5C) is shown to visualize the temporal importance of each feature. Furthermore, the users can select the individual cases in the Projection View (Fig. 4.5E) and all the selected individual cases are grouped by similarity and displayed in the Individual View (Fig. 4.5D).



Figure 4.5. Design of Hidden unit distribution and feature glyph. A) Hidden unit cluster; B) Hidden unit distribution; C) Feature cluster; D) Feature distribution for selected features; E) Feature glyph design; G) Response link MultiRNNExplorer contains multiple coordinated views to support exploring and understanding RNNs' behaviors on multi-dimensional time-series data, especially on hidden unit response and feature importance. The Configuration Panel (B) allows users to select a RNN models and configure parameters. To reveal model mechanism, the Cluster View (A) summarizes the hidden unit clusters' response to feature clusters, and the Feature Importance View (C) summarizes the temporal importance of input features. The Projection View (E) displays a data overview, allowing users to identify and select sequence instances of interest for further analysis. The selected instances will be shown by the Individual View (D).

#### 4.6.1 Cluster View

The Cluster View (Fig. 4.5A) shows the overview of response relationship (**T3**) between the hidden units and features. The hidden units and features are visualized as the Hidden State Distribution and the Feature Glyph respectively.

**Hidden State Distribution.** The left column on the Cluster View is the Hidden State Distribution component. As shown in Fig. 4.6A, each row represents a hidden unit cluster. Each hidden unit in a cluster is represented as a line chart that shows its activation distribution(**T1**). The x-axis represents the hidden unit output ranging from  $-1$  to  $+1$  and the y-axis represents the corresponding probability (Fig. 4.6B). From the line chart, users can observe and compare the activation distribution patterns of different hidden units.

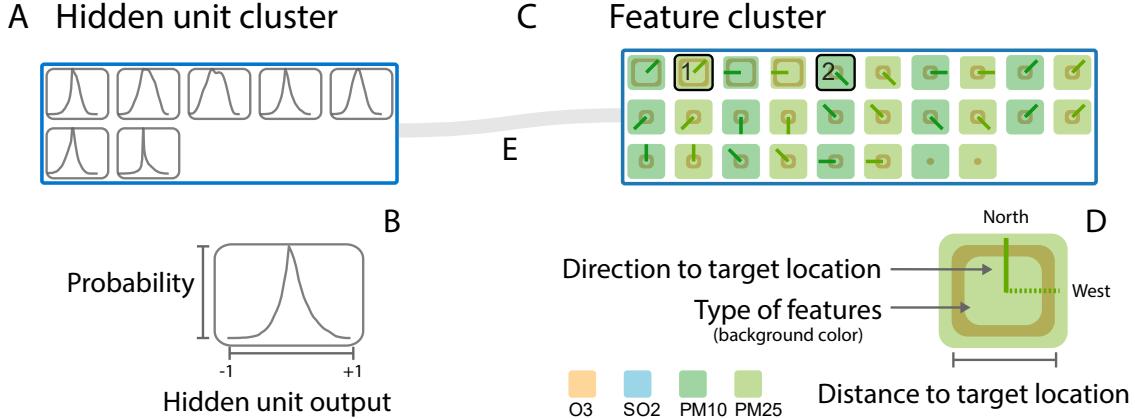


Figure 4.6. Design of Hidden unit distribution and feature glyph. A) Hidden unit cluster; B) Hidden unit distribution; C) Feature cluster; D) Feature distribution for selected features; E) Feature glyph design; G) Response link

**Feature Glyph.** The right column of the Cluster View is the Feature Glyph component (Fig. 4.6C). Similar to the Hidden State Distribution, each row represents a feature cluster in which a glyph (Fig. 4.6D) represents a feature. As described in Sec. 4.3.1, we define our usage scenario as air pollution forecasting. Each feature has three identifiers: the feature category, the direction, and the distance from the feature to the target location. The background color of the feature glyph cell encodes the feature category. We use a categorical color scheme to encode different categories, and users can find the color legend at the top of the Cluster View. In each feature glyph, the distance and direction to the target location are presented by a rectangle and a line segment with one end point at the center. The angle of the direction bar encodes the direction, and the width of the distance rectangle represents the distance(Fig. 4.6D).

**Interactions.** We also support various interactions to allow users to dynamically explore this view. The curves linking the hidden state cluster and feature cluster with the width indicate the response strength (Fig. 4.6E). Users can also filter the link according to the response strength by adjusting the slider bar. When hovering over a hidden state cluster or a feature cluster, the corresponding links and linked clusters will be highlighted.

In this view, the users can obtain an overview of the response relationship between hidden units and features, for example, we can find that there are not strong links connecting to cluster 8 (Fig. 4.5 A<sub>3</sub>), this may be because that all the hidden units are “weakly” activated in cluster 8.

## 4.6.2 Feature Importance View

The feature importance view allows users to explore the feature contribution to model output (**T2**). As discussed in Sec.4.5.3, with an input case, we are able to measure the importance of a single feature as a sequence of importance scores which correspond to the importance at all timestamps (Fig. 4.5C).

Since the importance score only provides a local description for the feature importance, an effective visualization is needed to show an overview of each feature’s importance. We choose boxplot for this task since it can present the statistics overview. To show the temporal trend of a feature, we group the importance score of all test cases by the timestamps and make statistics group by group. For the test sequence with a length  $T$ , will use the feature importance charts which contain  $T$  boxplots to show the trend of feature importance (Fig. 4.5C).

The horizontal axis indicates the timestamps and the vertical axis indicates the feature importance score. The top line, upper edge, middle line, bottom edge and bottom line of the boxplot indicate the maximum, 75<sup>th</sup> percentile, mean, 25<sup>th</sup> percentile and minimum of the importance scores. Since sometimes the maximum will much larger than the 75<sup>th</sup> percentile value, which makes the box vary flat and difficult for users to explore the temporal pattern, we limit the maximum score  $Ms$  shown in the view. If a boxplot has scores larger than  $Ms$ , a diamond symbol appears on the top of the boxplot. The opacity of the diamond indicates the magnitude of the absolute difference between the largest score and  $Ms$ .

We also define the overall importance score for a single feature as the sum of the mean score at all timestamps. By default, the boxplot charts will be ranked according to the overall feature importance score. Due to the large number of features, only the top 10 feature importance charts are visualized. Users may observe other features by using the scroll bar or filtering the features from the projection view (Fig. 4.5C).

## 4.6.3 Projection View

To help users obtain an overview of case clusters and outliers (**T5**), we design the Projection View (Fig. 4.5E) which supports various interactions such as zooming and brushing to allow users to select a subset of data for further examination.

In the Projection View, each circle represents a individual case. There are many multi-dimensional

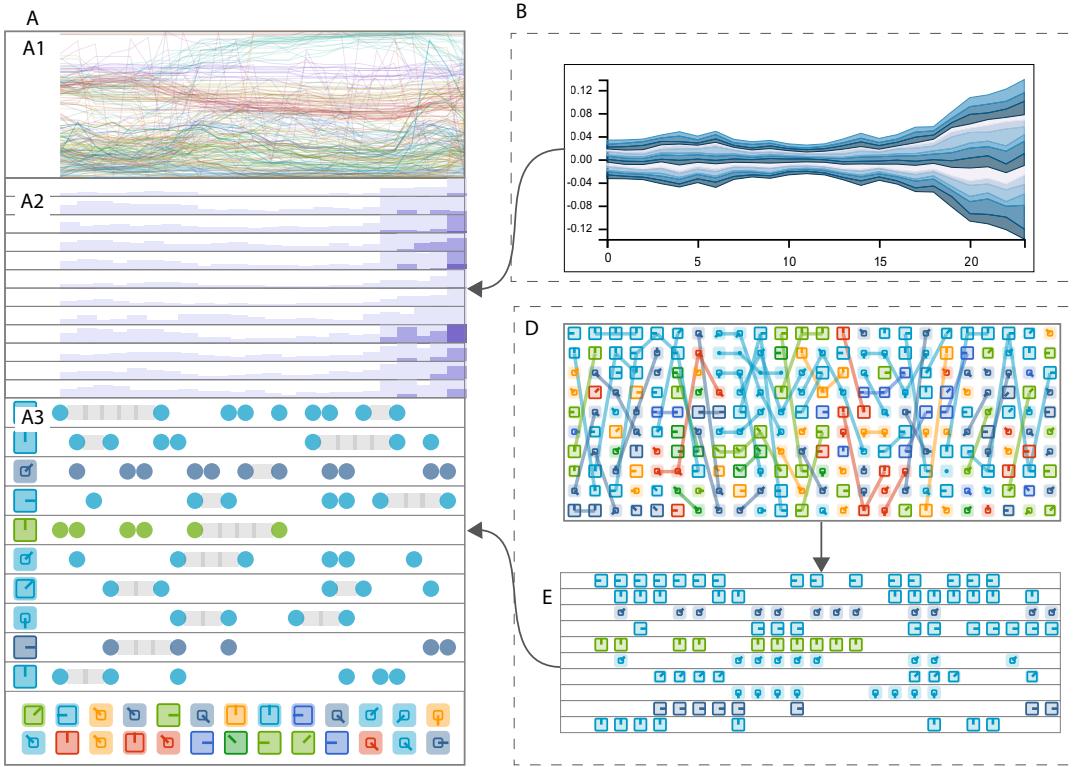


Figure 4.7. Individual design and the alternative designs. A) Individual View. A1) Feature Trend Chart; A2) Cluster Importance Chart; A3) Top Features Chart. B) themeriver as the alternative design of the Cluster Importance Chart; C) and D) node-like sequence and node sequence as the alternative design of Top Features List.

reduction methods such as MDS and PCA; we select t-SNE as it can strongly repel dissimilar points and show clusters clearly. For each case, we collect the feature cluster importance over all time steps (discussed in Sec.4.5.3) as the input vectors of t-SNE. Thus, the positions of the circles reflect the similarity of their cluster importance.

Furthermore, to improve the flexibility of the case selection, we add a two-scale timeline (Fig. 4.5E top) to show the target feature trend, enabling user filtering of the cases by time, and a feature selection component (Fig. 4.5E left) to filter the cases by feature value.

#### 4.6.4 Individual View

After observing an overview on data similarities, users may need to drill down to a few individual cases of interest for detailed examination. We develop the Individual View for users to explore and compare the different individuals over time (**T6**).

The selected individual cases are visualized as several stacks of cell as in Fig. 4.7A. Each cell consists of three components: the Feature Trend Chart (Fig. 4.7A<sub>1</sub>), the Cluster Importance Chart (Fig. 4.7A<sub>2</sub>), and the Top Feature List (Fig. 4.7A<sub>3</sub>). All these three components share the same x-axis which represents the time where time steps increase from left to right.

Feature Trend Chart is a multi-line chart that depicts how different features' values change over time. The y-axis represents normalized feature values where the feature value increases from bottom to top ranging between 0 and 1. Each line represents a feature and the line color encodes feature category the same as in the Cluster View. The corresponding lines are highlighted when hovering on any feature glyph in the Cluster View or hovering on feature importance view.

The middle component is the Cluster Importance Chart, which is a list of horizon bar charts that summarize how each feature group's gradient changes over time. Each feature group is represented as a horizon bar chart and aligned vertically in the same order as the Cluster View. For a horizon bar chart, each bar represents the averaged gradient for the corresponding feature group at one time step. We use both the bar height and bar color to encode the gradient value. As shown in Fig.4.7A<sub>3</sub>, the first visual channel to encode gradient value is bar height where a greater height indicates a larger gradient value. When the gradient value exceeds a certain limit, we clip the bar and overlay another darker bar with the same height as the clipped part at the same position for better vertical space efficiency. We have also considered other design choices such as a themeriver (Fig.4.7B) in which each colored flow indicates a feature group. However, comparing different feature groups may be difficult and it requires more space when the gradient is large. Thus, we abandon this alternative choice and adopt the current design.

Though the Cluster Importance Chart provides an overview of how each feature cluster's importance changes over time, users still need to link this component to the Cluster View to observe which features are considered important by the model. We design a Top Feature List to visualize the important features over time. Our first design is shown in Fig. 4.7D. where the top  $N$  features ranked by importance are visualized as feature glyphs at each timestamp. We draw links to connect glyphs that represent the same feature in consecutive time steps. However, this design leads to serious visual clutter due to link overlap when feature ranks are frequently changed over time. To alleviate users' mental burden, we develop a new design as shown in Fig. 4.7E. Each feature is represented as a row and we draw its corresponding feature glyph at the timestamps when this feature is ranked in the top  $N$  most important features. To simplify this visual design, we position

the feature glyph at the beginning of each row. We then use two colored circles linked with gray lines to indicate at which time steps the corresponding feature is ranked in the top  $N$  mostly important features. The circle color is consistent with the feature glyph color. To make the Top Feature List space efficient, we only show ten rows by default, and other features are collapsed as feature glyph rows as shown in the bottom at Fig.4.7A<sub>3</sub>. Users can click the glyph rows to select different features to analyze.

The three components enable users to observe which features are considered important by the model over time and how the importance is related to feature value changes. Users can also append multiple cells to the Sequence View to compare different sequences side by side. When the number of cells becomes large, we use dbSCAN to cluster the similar individual cases by the fatten cluster importance(discussed in sec. 4.5.3) into one stacked cells with one randomly selected case as the representative case at the top of stack.

#### 4.6.5 Interactions and Linkage

To better facilitate the interactive exploration of RNNs, our system supports cross-view interactions.

**Cross-view highlight.** There are three key visualization components appearing across different views: **features**, **feature clusters**, and **cases**. These components are visualized and encoded in different approaches in across views to support various analysis requirements. If one feature is selected, its corresponding visual elements in other views will be highlighted.

**Linkage between individual view and feature importance view.** When multiple individual cases are selected, the feature importance by time will be visualized as multi-line charts as Fig. 4.5C shows. When users hovering over an individual case , the corresponding line-chart will be highlighted.

### 4.7 Case study

In this section, we demonstrate the effectiveness of MultiRNNExplorer in analyzing model behaviors and feature importance. We use the air pollutant data between 2015 to 2017 to train the model and use 8,375 cases in 2018 as testing data for analysis. The accuracy and hyper-parameters of dif-

ferent models are listed in Table 4.2. We demonstrate our system to the domain expert and analyze the trained models on several tasks.

Model	Size	Dense Layer	MSE ( $PM_{2.5}$ )
Vanilla RNN	100	No	$5.31 \pm 0.98$
GRU	100	No	$4.32 \pm 0.51$
LSTM	100	No	$4.81 \pm 0.31$
GRU-Dense	100	3	$4.25 \pm 0.21$
LSTM-Dense	100	3	$4.53 \pm 0.53$

Table 4.2. Configuration and performance of RNNs, including vanilla RNN, GRU, LSTM, and the RNNs with dense layer (e.g., RNN-Dense). The performance is evaluated by the mean square error (MSE) of  $PM_{2.5}$ ; low MSE represents better performance.

## 4.8 Discussion and Conclusion

In this chapter, we presented MultiRNNExplorer, a visual analytic system for understanding RNN models for high-dimensional time-series forecasting. Specifically, we use air pollutant forecasting as the target application. To understand the the model mechanism from a global perspective, we propose a technique to estimate the hidden units' response to an individual feature by measuring how different feature selections affect the hidden units' output distribution. From a finer granularity, we further use the gradient-based method to measure the local feature importance for each sequence instance. Based on these techniques, we design a visual analytic system which enables the users to explore and reason about the model behavior from different perspectives. Our evaluation includes three case studies that demonstrate the effectiveness of the proposed system for comprehensive analysis of RNNs. Meanwhile, there are some issues need to be discussed:

**Scalability.** Several views may suffer scalability issues when the number of cases increase. In Projection View, thousands of individual cases need to be visualized and cause serious visual clutter due to the overlap of circles. In our case, more than 8000 points are visualized. If data size keeps increasing, we may also apply other advanced projection techniques [92, 114] for Projection View. The individual view also has such a problem: it is easy for users to brush hundreds of individual cases from Projection View and generate tens of clusters. Due to the limited screen size, our current design allows 9 groups of individual cases to be shown at the same time and uses the scroll bar to enable the observation of more groups.

**Generalization.** Though we use air pollutant forecasting as example in this chapter, the proposed method can be easily extended to other high-dimensional time-series forecasts with few

changes. The only design that may need to be fine-tuned to is feature glyphs (Fig. 4.6). The current design supports encoding three spatial attributes including direction, type, and distance and more design choices can be explored according to different requirements in other domains.

There are also some future directions to improve MultiRNNExplorer. One approach is to improve the individual comparison. In our current design, the individual comparison requires comparing data instances side by side. Supporting interactions to highlight the differences would further benefit users in this scenario. We also consider to use our system on other high-dimensional forecasting applications including such as fraud detection [59]. We are also exploring the possibility of applying MultiRNNExplorer to audio modeling tasks.

## CHAPTER 5

### CONCLUSION AND FUTURE WORK

The fast increasing global urbanization process in recent years has greatly changed the world in the recent 50 years. The human migration from the rural area to the urban area results in a series of problems such as environment pollutant, traffic jam and insufficient local resource supply. Meanwhile, with the development of science and technology, a variety of data source can be collected easier and cheaper, which also provide the opportunity for analysts to understand the phenomenon and solve the problems. Although a lot of automatic techniques have been proposed in recent years, most of them still have limitations such as the lack of interpretability, difficulty in handling unexpected cases. Due to the variety of urban applications, these techniques always generate uncontrollable results which is hard to be accepted by the non-expert users. Thus the human experience and knowledge are required to make judgement or provide guidance in the analyzing process. Visual analytics, integrating the intuitive graphics and auto techniques, bridge the gap between users and complex urban data by taking human into the analysis loop.

This proposal consists of three works that cover the three domains of urban information: place, people and technology. We start with a brief summary of the existing visualization techniques used in urban data exploration. After that, we introduced three visual analytics system target at the human-scale urban form exploration, large scale movement visualization and urban computing technique interpretation.

In Chapter 2, we introduce Streetvizor, a framework enabling urban analysts to explore the human scale urban from city-, region- and street-level. The proposed system takes the street view images as input and extracts human-scale urban form features by leveraging the advanced machine learning techniques. Then, a visual analytics system is built upon the urban form feature database. The proposed method is evaluated by the real world data from four cities: London, Singapore, Hong Kong, and New York City.

In Chapter 3, we have developed RAEB(route aware edge bundling), a novel edge bundling technique to visualize the overview of a large amount of OD trails in urban traffic data. Target at

the limitation of traditional edge bundling techniques, we introduce the urban traffic network as a constraint to the trail bundles. A series of new parameters are employed in the pipeline.

In Chapter 4, we propose MultiRNNExplorer, a visual analytics system which helps analysts to understand the RNNs in high-dimensional time series forecast. The visual analytics system aims to interpret RNNs from two aspects, the overall model mechanism and the feature importance. Several case studies on air pollutant forecast show the effectiveness of the proposed technique.

Even though several novel designs and comprehensive analytics systems have been discussed in this proposal, the current development of urban related visualization techniques still cannot meet the analysis requirement due to the increasing complexity and variety of the applications. Based on the current work, we list several promising future directions.

**High-level urban analysis based on street view images.** In Chapter 2, we discuss how to leverage the easily collected street view images and advanced machine learning techniques to help us build the comprehensive visual analytics system for human-scale urban form explorations. In the future, we will extend the current framework to the more complex tasks. For example, an exiting research work uses vehicle classification techniques to estimate the demographic makeup of the US; We can also extract the signboards and building style from the street view images to support the exploration of urban region functionalities.

**Improve the bundling and rendering efficiency for the large trajectory dataset.** In Chapter 3, the edge bundling and rendering process still require tens of seconds to finish the visualization tasks, which makes the interactive exploration impossible. Our proposed technique is built upon the kernel density estimation techniques which can be parallelized. In the future, we will use the GPU-acceleration to improve bundling efficiency. In addition, to improve the computing efficiency, another strategy is to reduce the data amount. Traditional sampling methods, such as uniform and stratified sampling, cannot keep visualization accuracy. We plan to develop visualization aware sampling techniques to support the fast OD trails bundling.

**Support model ensemble exploration.** Due to the complexity and variety of urban information exploration, a single forecast technique is always not enough to produce a credible and accurate result. In the future, we will extend the technique interpretation from RNNs to technique ensembles. However, use the same criteria to interpret multiple techniques is difficult due to the different algorithm mechanisms. Thus, we plan to figure out some uniform properties of these techniques (such as the feature importance) to make the them comparable.

## REFERENCES

- [1] Pierre Accorsi, Nathalie Lalande, Mickael Fabrègue, Agnès Braud, Pascal Poncelet, Arnaud Sallaberry, Sandra Bringay, Maguelonne Teisseire, Flavie Cernesson, and Florence Le Ber. Hydroqual: Visual analysis of river water quality. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 123–132. IEEE, 2014.
- [2] Robert Andrews, Joachim Diederich, and Alan B Tickle. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-based Systems*, 8(6):373–389, 1995.
- [3] Gennady Andrienko, Natalia Andrienko, Peter Bak, Daniel Keim, and Stefan Wrobel. *Visual analytics of movement*. 2013.
- [4] Gennady Andrienko, Natalia Andrienko, Wei Chen, Ross Maciejewski, and Ye Zhao. Visual analytics of mobility and transportation: State of the art and further research directions. *IEEE Transactions on Intelligent Transportation Systems*, 18(8):2232–2249, 2017.
- [5] Natalia Andrienko and Gennady Andrienko. Spatial generalization and aggregation of massive movement data. *IEEE Transactions on Visualization and Computer Graphics*, 17(2):205–219, 2011.
- [6] Natalia Andrienko and Gennady Andrienko. Visual analytics of movement: An overview of methods, tools and procedures. *Information Visualization*, 12(1):3–24, 2012.
- [7] Dragomir Anguelov, Carole Dulong, Daniel Filip, Christian Frueh, Stéphane Lafon, Richard Lyon, Abhijit Ogale, Luc Vincent, and Josh Weaver. Google street view: Capturing the world at street level. *Computer*, 43(6):32–38, 2010.
- [8] Sean M Arietta, Alexei A Efros, Ravi Ramamoorthi, and Maneesh Agrawala. City forensics: Using visual elements to predict non-visual city attributes. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2624–2633, 2014.

- [9] B. Bach, N. H. Riche, C. Hurter, K. Marriott, and T. Dwyer. Towards unambiguous edge bundling: Investigating confluent drawings for network visualization. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):541–550, 2017.
- [10] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint*, abs/1511.00561, 2015.
- [11] Roger Beecham, Jo Wood, and Audrey Bowman. Studying commuting behaviours using collaborative visual analytics. *Computers, Environment and Urban Systems*, 47:5–15, 2014.
- [12] Dirk Brockmann, Lars Hufnagel, and Theo Geisel. The scaling laws of human travel. *Nature*, 439(7075):462–465, 2006.
- [13] J. Böttger, A. Schäfer, G. Lohmann, A. Villringer, and D. S. Margulies. Three-dimensional mean-shift edge bundling for the visualization of functional connectivity in the brain. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):471–480, 2014.
- [14] Lee Byron and Martin Wattenberg. Stacked graphs—geometry & aesthetics. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1245–1252, 2008.
- [15] Qing Cao, Bradley T Ewing, and Mark A Thompson. Forecasting wind speed with recurrent neural networks. *European Journal of Operational Research*, 221(1):148–154, 2012.
- [16] Haidong Chen, Wei Chen, Honghui Mei, Zhiqi Liu, Kun Zhou, Weifeng Chen, Wentao Gu, and Kwan-Liu Ma. Visual abstraction and exploration of multi-class scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1683–1692, 2014.
- [17] Siming Chen, Xiaoru Yuan, Zhenhuang Wang, Cong Guo, Jie Liang, Zuchao Wang, Xiaolong Luke Zhang, and Jiawan Zhang. Interactive visual discovering of movement patterns from sparsely sampled geo-tagged social media data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):270–279, 2015.
- [18] Wei Chen, Fangzhou Guo, and Fei-Yue Wang. A survey of traffic data visualization. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):2970–2984, 2015.

- [19] Yingyi Chen, Qianqian Cheng, Yanjun Cheng, Hao Yang, and Huihui Yu. Applications of recurrent neural networks in environmental factor forecasting: A review. *Neural Computation*, 30(11):2855–2881, 2018.
- [20] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- [21] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (5):603–619, 2002.
- [22] D. Cornel, A. Konev, B. Sadransky, Z. Horváth, A. Brambilla, I. Viola, and J. Waser. Composite flow maps. *Computer Graphics Forum*, 35(3):461–470, 2016.
- [23] Mark Craven and Jude W Shavlik. Extracting tree-structured representations of trained networks. In *Advances in Neural Information Processing Systems*, pages 24–30, 1996.
- [24] Weiwei Cui, Hong Zhou, Huamin Qu, Pak Chung Wong, and Xiaoming Li. Geometry-based edge clustering for graph visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1277–1284, 2008.
- [25] Zikun Deng, Di Weng, Jiahui Chen, Ren Liu, Zhibin Wang, Jie Bao, Yu Zheng, and Yingcai Wu. Airvis: Visual analytics of air pollution propagation. *IEEE Transactions on Visualization and Computer Graphics*, 2019.
- [26] Inderjit S Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 269–274. ACM, 2001.
- [27] Somayeh Dodge, Robert Weibel, and Anna-Katharina Lautenschütz. Towards a taxonomy of movement patterns. *Information Visualization*, 7(3):240–252, 2008.
- [28] Carl Doersch, Saurabh Singh, Abhinav Gupta, Josef Sivic, and Alexei A Efros. What makes Paris look like Paris? *Commun. ACM*, 58(12):103–110, 2015.
- [29] Thomas Eiter and Heikki Mannila. *Computing Discrete Frechet Distance*. Technical Report CDTR 94/64, Christian Doppler Laboratory for Expert Systems. TU Vienna, Austria, 1994.

- [30] Ozan Ersoy, Christophe Hurter, Fernando Paulovich, Gabriel Cantareiro, and Alex Telea. Skeleton-based edge bundling for graph visualization. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2364–2373, 2011.
- [31] Nivan Ferreira, Marcos Lage, Harish Doraiswamy, Huy Vo, Luc Wilson, Heidi Werner, Muchan Park, and Cláudio Silva. Urbane: A 3d framework to support data driven decision making in urban development. In *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 97–104. IEEE, 2015.
- [32] Nivan Ferreira, Lauro Lins, Daniel Fink, Steve Kelling, Christopher Wood, Juliana Freire, and Claudio Silva. Birdvis: Visualizing and understanding bird populations. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2374–2383, 2011.
- [33] Nivan Ferreira, Jorge Poco, Huy T Vo, Juliana Freire, and Cláudio T Silva. Visual exploration of big spatio-temporal urban data: A study of New York city taxi trips. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2149–2158, 2013.
- [34] Marcus Foth, Jaz Hee-jeong Choi, and Christine Satchell. Urban informatics. In *Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work*, pages 1–8. ACM, 2011.
- [35] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.
- [36] Ying-Huey Fua, Matthew O Ward, and Elke A Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In *Proceedings of the Conference on Visualization'99: Celebrating Ten Years*, pages 43–50. IEEE Computer Society Press, 1999.
- [37] Timnit Gebru, Jonathan Krause, Yilun Wang, Duyun Chen, Jia Deng, Erez Lieberman Aiden, and Fei-Fei Li. Using deep learning and Google street view to estimate the demographic makeup of the US. *arXiv preprint arXiv:1702.06683*, 2017.
- [38] Jan Gehl. *Life between Buildings: Using Public Space*. Van Nostrand Reinhold, 1971.
- [39] Michael Gleicher, Danielle Albers, Rick Walker, Ilir Jusufi, Charles D Hansen, and Jonathan C Roberts. Visual comparison for information visualization. *Information Visualization*, 10(4):289–309, 2011.

- [40] Sarah Goodwin, Jason Dykes, Aidan Slingsby, and Cagatay Turkay. Visualizing multiple variables across scale and geography. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):599–608, 2015.
- [41] Google. Street View Image API. <https://developers.google.com/maps/documentation/streetview/>. Accessed: 2017-03-20.
- [42] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, 2017.
- [43] Joachim Gudmundsson, Patrick Laube, and Thomas Wolle. Computational movement analysis. In *Springer Handbook of Geographic Information*, pages 423–438. 2011.
- [44] Diansheng Guo. Flow mapping and multivariate visualization of large spatial interaction data. *IEEE Transactions on Visualization and Computer Graphics*, 15(6), 2009.
- [45] Diansheng Guo and Xi Zhu. Origin-destination flow data smoothing and mapping. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2043–2052, 2014.
- [46] Susan Havre, Elizabeth Hetzler, Paul Whitney, and Lucy Nowell. Themeriver: Visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):9–20, 2002.
- [47] Julian Heinrich and Daniel Weiskopf. State of the art of parallel coordinates. In *Eurographics - State of The Art Report*, pages 95 – 116, 2013.
- [48] Karl Moritz Hermann, Tomas Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015.
- [49] Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, 91(1), 1991.
- [50] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

- [51] Danny Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, 2006.
- [52] Danny Holten and Jarke J van Wijk. Force-directed edge bundling for graph visualization. *Computer Graphics Forum*, 28(3):983–990, 2009.
- [53] Danny Holten and Jarke J Van Wijk. Evaluation of cluster identification performance for different pcp variants. In *Computer Graphics Forum*, volume 29, pages 793–802. Wiley Online Library, 2010.
- [54] Xiaoke Huang, Ye Zhao, Chao Ma, Jing Yang, Xinyue Ye, and Chong Zhang. Trajgraph: A graph-based visual analytics approach to studying urban network centralities using taxi trajectory data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):160–169, 2015.
- [55] C. Hurter, S. Puechmorel, F. Nicol, and A. Telea. Functional decomposition for bundled simplification of trail sets. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):500–510, 2018.
- [56] Christophe Hurter, Ozan Ersoy, and Alexandru Telea. Graph bundling by kernel density estimation. *Computer Graphics Forum*, 31(3pt1):865–874, 2012.
- [57] Jane Jacobs. *The Life and Death of Great American Cities*. Random House, New York, 1961.
- [58] Bin Jiang, Chun-Yen Chang, and William C. Sullivan. A dose of nature: Tree cover, stress reduction, and gender differences. *Landscape Urban Plan.*, 132:26–36, 2014.
- [59] Johannes Jurgovsky, Michael Granitzer, Konstantin Ziegler, Sylvie Calabretto, Pierre-Edouard Portier, Liyun He-Guelton, and Olivier Caelen. Sequence classification for credit-card fraud detection. *Expert Systems with Applications*, 100:234–245, 2018.
- [60] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*, 2015.
- [61] R. Krüger, G. Simeonov, F. Beck, and T. Ertl. Visual interactive map matching. *IEEE Transactions on Visualization and Computer Graphics*, 24(6):1881 – 1892, 2018.

- [62] Robert Krüger, Dennis Thom, Michael Wörner, Harald Bosch, and Thomas Ertl. TrajectoryLenses – a set-based filtering and exploration technique for long-term trajectory data. *Computer Graphics Forum*, 32(3pt4):451–460, 2013.
- [63] Robert Krüger, Dennis Thom, Michael Wörner, Harald Bosch, and Thomas Ertl. Trajectorylenses—a set-based filtering and exploration technique for long-term trajectory data. In *Computer Graphics Forum*, volume 32, pages 451–460. Wiley Online Library, 2013.
- [64] Bum Chul Kwon, Min-Je Choi, Joanne Taery Kim, Edward Choi, Young Bin Kim, Soonwook Kwon, Jimeng Sun, and Jaegul Choo. Retainvis: Visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):299–309, 2019.
- [65] O. H. Kwon, C. Muelder, K. Lee, and K. L. Ma. A study of layout, rendering, and interaction methods for immersive graph visualization. *IEEE Transactions on Visualization and Computer Graphics*, 22(7):1802–1815, 2016.
- [66] A. Lambert, R. Bourqui, and D. Auber. 3D edge bundling for geographical data visualization. In *Proceedings of International Conference Information Visualisation*, pages 329–335, 2010.
- [67] Antoine Lambert, Romain Bourqui, and David Auber. Winding roads: Routing edges into bundles. *Computer Graphics Forum*, 29(3):853–862, 2010.
- [68] Joel Lanir, Peter Bak, and Tsvi Kuflik. Visualizing proximity-based spatiotemporal behavior of museum visitors using tangram diagrams. In *Computer Graphics Forum*, volume 33, pages 261–270. Wiley Online Library, 2014.
- [69] Antoine Lhuillier, Christophe Hurter, and Alexandre Telea. State of the art in edge and trail bundling techniques. *Computer Graphics Forum*, 36(3):619–645, 2017.
- [70] Antoine Lhuillier, Christophe Hurter, and Alexandru Telea. FFTEB: Edge bundling of huge graphs by the fast fourier transform. In *Proceedings of the IEEE Pacific Visualization Symposium (PacificVis)*, pages 190–199, 2017.

- [71] Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066*, 2015.
- [72] Xiang Li, Ling Peng, Xiaojing Yao, Shaolong Cui, Yuan Hu, Chengzeng You, and Tianhe Chi. Long short-term memory neural network for air pollutant concentration predictions: Method development and evaluation. *Environmental Pollution*, 231:997–1004, 2017.
- [73] Xiaojiang Li, Chuanrong Zhang, Weidong Li, Robert Ricard, Qingyan Meng, and Weixing Zhang. Assessing street-level urban greenery using Google street view and a modified green view index. *Urban For. Urban Gree.*, 14(3):675–685, 2015.
- [74] Lauro Lins, James T Klosowski, and Carlos Scheidegger. Nanocubes for real-time exploration of spatiotemporal datasets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2456–2465, 2013.
- [75] Zachary C Lipton. The doctor just won’t accept that! *arXiv preprint arXiv:1711.08037*, 2017.
- [76] Dongyu Liu, Di Weng, Yuhong Li, Jie Bao, Yu Zheng, Huamin Qu, and Yingcai Wu. Smar-tadp: Visual analytics of large-scale taxi trajectories for selecting billboard locations. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):1–10, 2016.
- [77] Mengchen Liu, Jiaxin Shi, Zhen Li, Chongxuan Li, Jun Zhu, and Shixia Liu. Towards better analysis of deep convolutional neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):91–100, 2017.
- [78] Xingjian Liu, Yan Song, Kang Wu, Jianghao Wang, Dong Li, and Ying Long. Understanding urban China with open data. *Cities*, 47:53 – 61, 2015.
- [79] Ying Long and Liu Liu. How green are the streets? an analysis for central areas of Chinese cities using Tencent street view. *PLOS ONE*, 12(2):1–18, 02 2017.
- [80] Ying Long and Yu Ye. Human-scale urban form: Measurements, performances, and urban planning & design interventions. *South Architecture*, 36(5):39 – 45, 2016.
- [81] Yin Lou, Chengyang Zhang, Yu Zheng, Xing Xie, Wei Wang, and Yan Huang. Map-matching for low-sampling-rate gps trajectories. In *Proceedings of ACM SIGSPATIAL In-*

- ternational Conference on Advances in Geographic Information Systems*, pages 352–361, 1653820, 2009.
- [82] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017.
- [83] Sheng-Jie Luo, Chun-Liang Liu, Bing-Yu Chen, and Kwan-Liu Ma. Ambiguity-free edge-bundling for interactive graph visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18(5):810–821, 2012.
- [84] Frederik Maes, Andre Collignon, Dirk Vandermeulen, Guy Marchal, and Paul Suetens. Multimodality image registration by maximization of mutual information. *IEEE Transactions on Medical Imaging*, 16(2):187–198, 1997.
- [85] Yao Ming, Shaozu Cao, Ruixiang Zhang, Zhen Li, Yuanzhe Chen, Yangqiu Song, and Huamin Qu. Understanding hidden memories of recurrent neural networks. In *2017 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 13–24. IEEE, 2017.
- [86] Nikhil Naik, Jade Philipoom, Ramesh Raskar, and César Hidalgo. Streetscore-predicting the perceived safety of one million streetscapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 779–785, 2014.
- [87] Bing Ni, Qiaomu Shen, Jiayi Xu, and Huamin Qu. Spatio-temporal flow maps for visualizing movement and contact patterns. *Visual Informatics*, 1(1):57–64, 2017.
- [88] OpenStreetMap. OSM Wiki. [http://wiki.openstreetmap.org/wiki/Main\\_Page](http://wiki.openstreetmap.org/wiki/Main_Page). Accessed: 2017-03-20.
- [89] Mihaela Oprea, Marian Popescu, and Sanda Florentina Mihalache. A neural network based model for pm 2.5 air pollutant forecasting. In *2016 20th International Conference on System Theory, Control and Computing (ICSTCC)*, pages 776–781. IEEE, 2016.
- [90] Thomas Ortner, Johannes Sorger, Harald Steinlechner, Gerd Hesina, Harald Piringer, and Eduard Gröller. Vis-a-ware: Integrating spatial and non-spatial visualization for visibility-aware urban planning. *IEEE Transactions on Visualization and Computer Graphics*, 23(2):1139–1151, 2016.

- [91] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.*, 12:2825–2830, 2011.
- [92] Nicola Pezzotti, Thomas Höllt, B Lelieveldt, Elmar Eisemann, and Anna Vilanova. Hierarchical stochastic neighbor embedding. In *Computer Graphics Forum*, volume 35, pages 21–30. Wiley Online Library, 2016.
- [93] Nicola Pezzotti, Thomas Höllt, Jan Van Gemert, Boudewijn PF Lelieveldt, Elmar Eisemann, and Anna Vilanova. Deepeyes: Progressive visual analytics for designing deep neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):98–108, 2018.
- [94] Doantam Phan, Ling Xiao, Ron Yeh, and Pat Hanrahan. Flow map layout. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis)*, pages 219–224, 2005.
- [95] Huamin Qu, Wing-Yi Chan, Anbang Xu, Kai-Lun Chung, Kai-Hon Lau, and Ping Guo. Visual analysis of the air pollution problem in hong kong. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1408–1415, 2007.
- [96] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016.
- [97] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.
- [98] Andrew G Rundle, Michael DM Bader, Catherine A Richards, Kathryn M Neckerman, and Julien O Teitler. Using Google street view to audit neighborhood environments. *Am. J. Prev. Med.*, 40(1):94–100, 2011.
- [99] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.
- [100] David Selassie, Brandon Heller, and Jeffrey Heer. Divided edge bundling for directional network data. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2354–2363, 2011.

- [101] David Selassie, Brandon Heller, and Jeffrey Heer. Divided edge bundling for directional network data. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2354–2363, 2011.
- [102] Heng Shi, Minghao Xu, and Ran Li. Deep learning for household load forecasting—a novel pooling deep rnn. *IEEE Transactions on Smart Grid*, 9(5):5271–5280, 2017.
- [103] Ben Shneiderman. Tree visualization with Tree-maps: 2-d space-filling approach. *ACM Trans. Graph.*, 11(1):92–99, 1992.
- [104] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proc. IEEE Symp. Visual Languages*, pages 336–343, 1996.
- [105] Hendrik Strobelt, Sebastian Gehrmann, Michael Behrisch, Adam Perer, Hanspeter Pfister, and Alexander M Rush. Seq2seq-vis: A visual debugging tool for sequence-to-sequence models. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):353–363, 2019.
- [106] Hendrik Strobelt, Sebastian Gehrmann, Hanspeter Pfister, and Alexander M Rush. Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):667–676, 2018.
- [107] Guodao Sun, Yingcai Wu, Ronghua Liang, and Shixia Liu. A survey of visual analytics techniques and applications: State-of-the-art research and future challenges. *J. Comput. Sci. Technol.*, 28(5):852–867, 2013.
- [108] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deeply learned face representations are sparse, selective, and robust. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2892–2900, 2015.
- [109] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.
- [110] Matthias Thöny and Renato Pajarola. Vector map constrained path bundling in 3D environments. In *Proceedings of the ACM SIGSPATIAL International Workshop on GeoStreaming*, pages 33–42, 2015.
- [111] Waldo R. Tobler. Experiments in migration mapping by computer. *The American Cartographer*, 14(2):155–163, 1987.

- [112] Cagatay Turkay, Aidan Slingsby, Helwig Hauser, Jo Wood, and Jason Dykes. Attribute signatures: Dynamic visual summaries for analyzing multivariate geographical data. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2033–2042, 2014.
- [113] Matthew van der Zwan, Valeriu Codreanu, and Alexandru Telea. CUBu: Universal real-time bundling for large graphs. *IEEE Transactions on Visualization and Computer Graphics*, 22(12):2550–2563, 2016.
- [114] Vincent van Unen, Thomas Höllt, Nicola Pezzotti, Na Li, Marcel JT Reinders, Elmar Eisemann, Frits Koning, Anna Vilanova, and Boudewijn PF Lelieveldt. Visual analysis of mass cytometry data by hierarchical stochastic neighbour embedding reveals rare cell types. *Nature Communications*, 8(1):1740, 2017.
- [115] Carlos A Vanegas, Daniel G Aliaga, Bedrich Benes, and Paul Waddell. Visualization of simulated urban spaces: inferring parameterized generation of streets, parcels, and aerial imagery. *IEEE Transactions on Visualization and Computer Graphics*, 15(3):424–435, 2009.
- [116] Kevin Verbeek, Kevin Buchin, and Bettina Speckmann. Flow map layout via spiral trees. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2536–2544, 2011.
- [117] Tatiana Von Landesberger, Felix Brodkorb, Philipp Roskosch, Natalia Andrienko, Gennady Andrienko, and Andreas Kerren. Mobilitygraphs: Visual analysis of mass mobility dynamics via spatio-temporal graphs and clustering. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):11–20, 2015.
- [118] Tatiana von Landesberger, Felix Brodkorb, Philipp Roskosch, Natalia Andrienko, Gennady Andrienko, and Andreas Kerren. MobilityGraphs: Visual analysis of mass mobility dynamics via spatio-temporal graphs and clustering. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):11–20, 2016.
- [119] Fei Wang, Wei Chen, Feiran Wu, Ye Zhao, Han Hong, Tianyu Gu, Long Wang, Ronghua Liang, and Hujun Bao. A visual reasoning approach for data-driven transport assessment on urban road. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 103–112, 2014.
- [120] Fei Wang, Wei Chen, Feiran Wu, Ye Zhao, Han Hong, Tianyu Gu, Long Wang, Ronghua Liang, and Hujun Bao. A visual reasoning approach for data-driven transport assessment on

urban roads. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 103–112. IEEE, 2014.

- [121] Pu Wang, Timothy Hunter, Alexandre M Bayen, Katja Schechtner, and Marta C González. Understanding road usage patterns in urban areas. *Scientific Reports*, 2:1001, 2012.
- [122] Zhe Wang, Nivan Ferreira, Youhao Wei, Aarthy Sankari Bhaskar, and Carlos Scheidegger. Gaussian cubes: Real-time modeling for visual exploration of large multidimensional datasets. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):681–690, 2016.
- [123] Zuchao Wang, Min Lu, Xiaoru Yuan, Junping Zhang, and Huub Van De Wetering. Visual traffic jam analysis based on trajectory data. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2159–2168, 2013.
- [124] Leland Wilkinson and Michael Friendly. The history of the cluster heat map. *The American Statistician*, 63(2):179–184, 2009.
- [125] Jo Wood, Jason Dykes, and Aidan Slingsby. Visualisation of origins, destinations and flows with od maps. *The Cartographic Journal*, 47(2):117–129, 2010.
- [126] Wenchao Wu, Jiayi Xu, Haipeng Zeng, Yixian Zheng, Huamin Qu, Bing Ni, Mingxuan Yuan, and Lionel M Ni. Telcovis: Visual exploration of co-occurrence in urban human mobility based on telco data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):935–944, 2015.
- [127] Shi Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, pages 802–810, 2015.
- [128] Panpan Xu, Yingcai Wu, Enxun Wei, Tai-Quan Peng, Shixia Liu, Jonathan JH Zhu, and Huamin Qu. Visual analysis of topic competition on social media. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2012–2021, 2013.
- [129] Xinsong Yang, Lei Shi, Madelaine Daianu, Hanghang Tong, Qingsong Liu, and Paul Thompson. Blockwise human brain network visual comparison using nodetrix representation. *IEEE transactions on visualization and computer graphics*, 23(1):181–190, 2016.

- [130] Wei Zeng, C. W. Fu, S. Müller Arisona, A. Erath, and H. Qu. Visualizing waypoints-constrained origin-destination patterns for massive transportation data. *Computer Graphics Forum*, 35(8):95–107, 2016.
- [131] Wei Zeng, Chi-Wing Fu, Stefan Müller Arisona, Simon Schubiger, Remo Burkhard, and Kwan-Liu Ma. Visualizing the relationship between human mobility and points of interest. *IEEE Transactions on Intelligent Transportation Systems*, 18(8):2271–2284, 2017.
- [132] Wei Zeng and Yu Ye. Vitalvizor: A visual analytics system for studying urban vitality. *IEEE Computer Graphics and Applications*, 38(5):38–53, 2018.
- [133] Xin Zhao and Arie Kaufman. Structure revealing techniques based on parallel coordinates plot. *Vis. Comput.*, 28(6):541–551, 2012.
- [134] Yixian Zheng, Wenchao Wu, Yuanzhe Chen, Huamin Qu, and Lionel M Ni. Visual analytics in urban computing: An overview. *IEEE Transactions on Big Data*, 2(3):276–296, 2016.
- [135] Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2267–2276. ACM, 2015.
- [136] Hong Zhou, Panpan Xu, Xiaoru Yuan, and Huamin Qu. Edge bundling in information visualization. *Tsinghua Science and Technology*, 18(2):145–156, 2013.
- [137] Hong Zhou, Xiaoru Yuan, Weiwei Cui, Huamin Qu, and Baoquan Chen. Energy-based hierarchical edge clustering of graphs. In *Proceedings of the IEEE Pacific Visualization Symposium (PacificVis)*, pages 55–61, 2008.
- [138] Hong Zhou, Xiaoru Yuan, Huamin Qu, Weiwei Cui, and Baoquan Chen. Visual clustering in parallel coordinates. In *Computer Graphics Forum*, volume 27, pages 1047–1054. Wiley Online Library, 2008.
- [139] Shanshan Zhou, Wenjing Li, and Junfei Qiao. Prediction of pm2. 5 concentration based on recurrent fuzzy neural network. In *2017 36th Chinese Control Conference (CCC)*, pages 3920–3924. IEEE, 2017.